

سلسلة ملخصات شوم

نظريات ومساائل في

البرمجة بلغة الكوبل

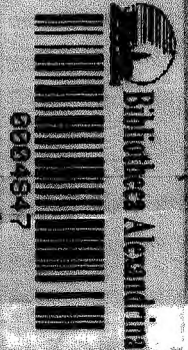
يتضمن الكوبل الهيكل

لورنس نيوكومر

يحتوي الكتاب على ٦٨٤ مسألة محلولة



الدار الدولية للنشر والتوزيع



ملخصات شوم
نظريات ومسائل
فى
البرمجة بلغة الكوبل

تأليف

لورنس نيوكومر

أستاذ علوم الحاسب المساعد
جامعة ولاية بنسلفانيا

ترجمة ومراجعة

د. مهنس / سرور على إبراهيم سرور

كلية الاقتصاد والإدارة
جامعة الملك سعود - فرع القصيم



الدار الدولية للنشر والتوزيع
القاهرة - الكويت - لندن

حقوق النشر

• الطبعة الأنجليزية : حقوق التأليف © ١٩٨٤ دار ماكجروهيل للنشر، إنك، جميع الحقوق محفوظة.

Schaum's Outline of theory & Problems of Programming with Structured Cobol

by

LAWRENCE R. NEWCOMER

• الطبعة العربية الأولى : حقوق الطبع والنشر © ١٩٩١، جميع الحقوق محفوظة للناشر

الدار الدولية للنشر والتوزيع

ص . ب ٥٥٩٩ هليوبوليس غرب - القاهرة

ت : ٢٥٨٢٨٨٧

تلكس : ٢٠٠٧٠ UN PBCRB

فاكس : ٢٩١٨٠٥٩ / ٠٠٢٠٢

لايجوز نشر أى جزء من هذا الكتاب أو أختزان مادته بطريقة الأسترجاع أو نقله على أى وجه أو بنى طريقة سواء كانت أليكترونية أو ميكانيكية أو بالتصوير أو خلاف ذلك إلا بموافقة الناشر على هذا كتابة مقدماً.

رقم الإيداع ١٩٩١/٣٤٥٨

I.S.B.N. 977-5107-26-1

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

« قُلْ لَوْ كَانَ الْبَحْرُ

مِدَاداً لِكَلِمَاتِ رَبِّي

لَنَفِدَ الْبَحْرُ قَبْلَ

أَنْ تَنْفَدَ كَلِمَاتُ رَبِّي،

وَلَوْ جِئْنَا بِمِثْلِهِ

مَدَداً »

حَقِّقْ اللَّهَ الْعَظِيمَ

مقدمة المؤلف

هذا الملخص يعلم الكويل (كما عرفه المعهد القومي الأمريكي للنمطيات عام 1974) * من وجهة نظر البرمجة المرتبة structured Programming ** فهو يغطي معظم ، وليس كل ، الاساليب التي تمكن المبرمجين من زيادة انتاجيتهم في اعداد برامج فردية . ونركز على استخدام تصميم الاجزاء من القمة الى القاعدة، مع تطبيق نمطيات متعددة من شفرة الكويل . ويهدف الكتاب إلى ان يكون شاملاً وموجزاً ، سامحاً للقارئ بالتقدم السريع . ومع نهاية الفصل الثاني يكون القارئ مُعداً لكتابة برامج بسيطة بلغة الكويل . وحيث إن تعلم برمجة الكمبيوتر يشبه تعلم السباحة (عليك بالقفز واداء العمل) فمطلوب من القارئ أن يتفد أكبر قدر ممكن من تمارين البرمجة مستخدماً جهاز كمبيوتر فعلي . وقد صمم الفصل التاسع ، الذي يغطي التصحيح ؛ بحيث يمكن قراءته في أى وقت بعد اتمام قراءة الفصل الثاني، وعليك أن ترجع إليه كلما وجدت مشكلة في تمارين البرمجة .

عند ظهور امتدادات لكويل 1974 النمطى (القياسى) فهي امتداد IBM OS/VS ، وسوف تعرف بهذا الاسم بوضوح *** . وكل الامثلة ما هي إلا مشاكل سبق اختبارها في جامعة ولاية بنسلفانيا الامريكية باستخدام نظام كمبيوتر IBM 3081 وكويل IBM OS/VS . وحيث إن المعلومات تختلف من نظام لآخر.. فقد قدمنا مادة مرتبطة تماماً بنظم من طراز IBM 370 ، مع نظم تشغيل من نوع OS/VE . ويعمل المعهد القومي الأمريكي للنمطيات حالياً على صيغة 1980 للكويل القياسى . والتغييرات المحتمل إدخالها في كويل الثمانينيات القياسى مقدمة في ملحق جـ **** .

المؤلف

* لقد صدر عن نفس المعهد صيغة مطورة من لغة الكويل عام ١٩٨٥ ، وسوف يتم التعرض لبعض السمات التي تميز هذه الصيغة عن الصيغة المعروفة عام ١٩٧٤ في ملحق جـ (المترجم)

** يطلق البعض اسم البرمجة الهيكلية أو البرمجة المتسلسلة على نفس الاسم إلا أن المترجم يفضل اسم البرمجة المرتبة لما له من وقع افضل لدى القارئ . (المترجم)

*** يجب التنويه أن هناك عديداً من صيغ الكويل الأخرى المتاحة تجارياً ومنها MS-COBOL

**** لقد ظهرت فعلاً صيغة الثمانينيات وهي كويل ١٩٨٥ النمطى وسوف يشتمل ملحق C على التغييرات الفعلية لهذه الصيغة عن صيغة الكويل ١٩٧٤ (المترجم)

مقدمة الناشر

المعرفة هي أصل الحضارة

والكلمة هي أصل المعرفة.

والكلمة المطبوعة هي أهم مكون في هذا المصدر.

وقد كانت الكلمة المطبوعة ولا تزال أهم وسائل الثقافة والأعلام وأوسعها إنتشاراً وأبقاها أثراً، حيث حملت إلينا حضارات

الأمم عبر السنين لتتولى الأجيال المتلاحقة صياغة حضارتها وإضاءة الطريق بنور العلم والمعرفة.

والكلمة تبقى مجرد فكرة لدى صاحبها حتى تتاح لها فرصة نشرها وترجمتها إلى لغات الآخرين، ثم توزيعها، وذلك وحده هو

الذي يكفل لها أداء رسالتها.

وعالم الكتب العلمية عالم رحب ممتد الآفاق، متسع الجنبات، والعلم لا وطن له ولا حدود، ويوم يحظى القارئ العربي بأحدث

الكتب العلمية باللغة العربية لهو اليوم الذي تتطلع له الأمة العربية جمعاء.

والدار الدولية للنشر والتوزيع تشعر بالرضا عن مساهمتها في هذا المجال بتقديم الطبعات العربية للكتب العلمية مستهدفة

توفير احتياجات القارئ العربي أستاذاً وباحثاً وممارساً.

والله ولي التوفيق..

محمد وفائي كامل

١٥	الفصل الأول : مفاهيم تشغيل البيانات	
١٥	١-١ كيف يعمل نظام الكمبيوتر	
١٦	٢-١ المدخلات	
١٦	٣-١ المخرجات	
١٧	٤-١ التخزين المساعد	
١٨	٥-١ الذاكرة	
١٩	٦-١ وحدة الحساب والمنطق ALU	
١٩	٧-١ وحدة التحكم CU	
٢٠	٨-١ لغات البرمجة	
٢٠	٩-١ نظم التشغيل	
٢١	١٠-١ الملفات	
٢٢	١١-١ شفرات البيانات	

٢٥	الفصل الثاني : عرض سريع للكوبل	
٢٧	١-٢ أجزاء الكوبل	
٢٩	٢-٢ الأقسام والمقاطع	
٣١	٣-٢ قواعد كتابة برامج كوبل	
٣٤	٤-٢ الاسماء التي يعرفها المبرمج ، والكلمات المحجوزة	
٣٥	٥-٢ الهدء	

٥١	الفصل الثالث : جزء التعريف	
٥١	١-٣ ترميز التكوين	
٥٢	٢-٣ تكوين جزء التعريف	
٥٣	٣-٣ مقطع تعريف البرنامج	

المحتوى

٥٧	الفصل الرابع : جزء الأوساط	
٥٧	١-٤ تكوين جزء الأوساط	١-٤
٥٩	٢-٤ قسم التشكيل : مقطع كمبيوتر المصدر	٢-٤
٥٩	٣-٤ قسم التشكيل : مقطع كمبيوتر الهدف	٣-٤
٦٠	٤-٤ قسم التشكيل : مقطع الأسماء الخاصة	٤-٤
٦٤	٥-٤ قسم المدخلات والمخرجات	٥-٤
٦٩	٦-٤ أمثلة على جزء الاوساط	٦-٤
٧٧	الفصل الخامس : جزء البيانات	
٧٧	١-٥ هيكل جزء البيانات	١-٥
٧٩	٢-٥ قسم الملفات	٢-٥
٨٠	٣-٥ وصف الملف وما تحتويه المجموعه	٣-٥
٨٤	٤-٥ وصف الملف وما يحتويه السجل	٤-٥
٨٥	٥-٥ وصف الملف وسجلات العناوين	٥-٥
٨٥	٦-٥ وصف الملف وسجلات البيانات	٦-٥
٨٧	٧-٥ وصف الملف والخطية	٧-٥
٨٩	٨-٥ وصف السجل فى قسم الملفات	٨-٥
٩٤	٩-٥ تكوين وصف البيانات	٩-٥
٩٦	١٠-٥ جزء الصورة	١٠-٥
١٠٧	١١-٥ جزء الاستخدام	١١-٥
١٠٩	١٢-٥ جزء الفراغ	١٢-٥
١١٠	١٣-٥ جزء التضييق	١٣-٥
١١١	١٤-٥ جزء الحدوث	١٤-٥
١١٢	١٥-٥ جزء الاشارة	١٥-٥
١١٣	١٦-٥ جزء التوافق	١٦-٥

الصفحة	المحتوى	الصفحة
--------	---------	--------

١١٤	جزء إعادة التعريف	١٧-٥
١١٦	تحديد طول عنصر البيانات	١٨-٥
١١٨	قسم مخزن العمل	١٩-٥
١٢٢	نطيات كتابة الشفرة	٢٠-٥

الفصل السادس : جزء الإجراءات

١٤١	مقدمة : نطيات كتابة الشفرة	١-٦
١٤٢	مدخلات / مخرجات : عبارة الفتح	٢-٦
١٤٥	مدخلات / مخرجات : عبارة الإغلاق	٣-٦
١٤٨	مدخلات : عبارة القراءة	٤-٦
١٥٠	مخرجات : عبارة الكتابة	٥-٦
١٥٨	مدخلات : عبارة القبول	٦-٦
١٦١	مخرجات : عبارة العرض	٧-٦
١٦٣	تشغيل : عبارة النقل	٨-٦
١٦٩	إيقاف تنفيذ البرنامج : عبارة التوقف	٩-٦
١٧٠	تنفيذ متحكم فيه لقطع : عبارة التنفيذ	١٠-٦
١٧٦	تشغيل : عبارة الجمع	١١-٦
١٧٩	تشغيل : حذف جزء من الكسر العشري وجزء التقريب	١٢-٦
١٨٠	تشغيل : السريان الزائد وجزء عند حدوث خطأ في الحجم	١٣-٦
١٨١	تشغيل : عبارة الطرح	١٤-٦
١٨٣	تشغيل : عبارة الضرب	١٥-٦
١٨٥	تشغيل : عبارة القسمة	١٦-٦
١٨٩	تشغيل : عبارة الحساب	١٧-٦
١٩٣	الكفاءة في الحسابات الرياضية	١٨-٦
١٩٤	نطيات كتابة شفرة اضافيه لجزء الإجراءات	١٩-٦

المسائل

٣٢٩	الفصل التاسع : التصحيح	
٣٢٩	١-٩ تصحيح الأخطاء التكوينية	
٣٣٢	٢-٩ تصحيح الأخطاء المنطقية الجسيمة	
٣٣٣	٣-٩ تصحيح الأخطاء المنطقية غير الجسيمة	
٣٣٣	٤-٩ الحصول على معلومات تتبع البرنامج	
٣٣٦	٥-٩ اخراج عناصر بيانات اثناء تنفيذ البرنامج	
٣٤٠	٦-٩ التوضيحات وعبارة يستخدم للتصحيح	

٣٥٧	الفصل العاشر : معالجة الجداول	
٣٥٧	١-١٠ جداول ذات بعد واحد : جزء الحدوث	
٣٥٨	٢-١٠ الدلائل	
٣٥٩	٣-١٠ معالجة جداول ذات بعد واحد	
٣٦٦	٤-١٠ جداول ذات بعدين	
٣٦٨	٥-١٠ معالجة جداول ذات بعدين	
٣٧٥	٦-١٠ جداول متغيرة الطول : جزء الحدوث طبقاً لـ	
٣٨٠	٧-١٠ الفهارس	
٣٨٥	٨-١٠ البحث المتتالي في الجداول وفعل إبحث	
٣٨٩	٩-١٠ البحث الثنائي في الجداول وفعل إبحث الكل	

٤٠٧	الفصل الحادي عشر : تشغيل الملفات تتابعياً	
٤٠٨	١-١١ تجديد الملفات التتابعية	
٤٠٩	٢-١١ خوارزمي خط الاتزان	
٤٢٧	٣-١١ التجديد في نفس المكان	

محتويات

٤٣٩	الفصل الثاني عشر : ترتيب ودمج الملفات	
٤٣٩	معجم ترتيب الملفات	١-١٢
٤٤٠	استخدام عبارة الترتيب	٢-١٢
٤٦٠	دمج الملفات بفعل الدمج	٣-١٢
٤٧٣	ملحق (أ) كلمات الكويل المحجوزة	
٤٨٧	ملحق (ب) تسلسل التتابع	
٤٩٥	ملحق (ج) الاختلافات عن كويل 1985 القياسي	
٥٠١	المصطلحات العلمية	

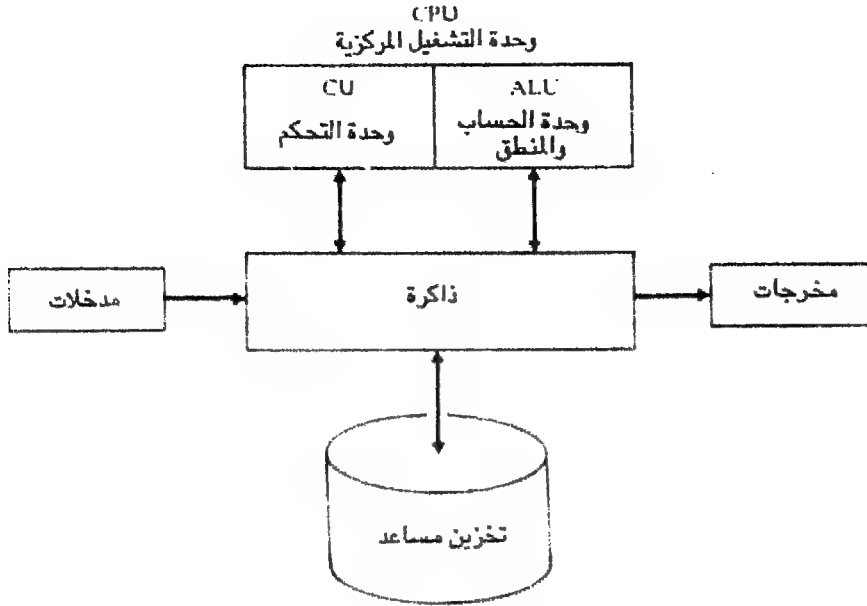
الفصل الأول

مفاهيم تشغيل البيانات

Data Processing Concepts

١-١ كيف يعمل نظام الكمبيوتر

يشار إلى الوحدات الطبيعية التي يتكون منها نظام الكمبيوتر بنظم المكونات hardware . ومن وجهة نظر المبرمج .. توجد للكمبيوتر مكونات وظيفية معينة في شكل (١-١)



شكل (١ - ١)

١ - ٢ المدخلات

وحدات المدخلات هي آلات تنقل معلومات من العالم الخارجى الى داخل ذاكرة الكمبيوتر . وفى معظم الاحوال .. علينا ان نضع المعلومات فى صورة مناسبة لاستخدامها بواسطة وحدة المدخلات أولا ، وعادة مايكون ذلك باحدى صور الطباعة . وتسمى المستندات التى تطبع منها المعلومات بمستندات المصدر Source documents . وحدات المدخلات الثلاث الرئيسية هي مايلى :

قارئات بطاقات

يستخدم مثقب بطاقات card punch اولا فى تسجيل المعلومات كتقوب فى بطاقات لها حجم نمطى . وتوضع مجموعة البطاقات بعد ذلك فى قارئ بطاقات، يشعر - بطريقة كهربائية ميكانيكية او بطريقة ضوئية ميكانيكية - بالتقوب الموجودة فى البطاقات، وينقل المعلومات الى ذاكرة الكمبيوتر .

من المفتاح إلى القرص

يشبه نظام من المفتاح الى القرص فى مفهومه نظام البطاقات ، إلا انه يسجل البيانات مغناطيسيا ، على قرص مرن floppy disk او على قرص صلب hard disk . عند ذلك يوضع القرص فى مشغل اقراص disk drive ، يستطيع الاحساس بالمعلومات المخزنة مغناطيسيا ونقلها الى ذاكرة الكمبيوتر . يقرأ مشغل الاقراص البيانات بسرعة اسرع كثيرا من قارئ البطاقات . وعلى عكس البطاقات.. فمن الممكن اعادة استخدام الاقراص المرنة أو الاقراص الصلبة لبيانات مختلفة .

نسخ دائمة ونهايات طرفية من أنبوب أشعة الكاثود

يمكن توصيل نهايات طرفية بالكمبيوتر بحيث تخدم لوحات مفاتيحها keyboards كوحدات مدخلات . وعند الضغط على احد المفاتيح.. ينقل الرمز لهذا المفتاح الى ذاكرة الكمبيوتر . والنهايات الطرفية الذكية intelligent terminals مشغل دقيق microprocessor (كمبيوتر دقيق على رقيقة متكاملة)، وبعض الذاكرة المبنية داخل النهاية الطرفية؛ بحيث تكون النهاية الطرفية قادرة على تخزين الرموز عند كتابتها ونقلها إلى الكمبيوتر ، على هيئة مجموعات (بدلاً من نقلها رمزاً رمزاً) . كما يمكن ان تجرى النهايات الطرفية الذكية بعض التشغيل المحلى للمعلومات داخل النهاية الطرفية نفسها . وكل من النهايات الطرفية الذكية والغبية (الصماء) dumb لها وحدات مخرجات كذلك كجزء من النهاية الطرفية :

للنهايات الطرفية للنسخ الدائمة hard copy طابعات تشبه الآلات الكاتبة، بينما لا يكون للنهايات الطرفية التى لها انبوب اشعة كاثود cathode-ray-tube (CRT) الا شاشة مرئية.

١ - ٣ المخرجات

تستقبل وحدات المخرجات output المعلومات من ذاكرة الكمبيوتر، وتنقلها إلى العالم الخارجى سواء كانت فى صورة مقروءة للإنسان human-readable أم فى صورة مقروءة لآلة machine-readable .

الطابعات

تسجل هذه الوحدات محتويات ذاكرة الكمبيوتر كرموز مطبوعة على ورق . وفى طابعات الطرق impact printers .. يضغط شريط الحبر على الورق، أما فى الطابعات التى لاتستخدم الطرق non impact printers .. فيمكن استخدام

وسائل حرارية او اليكتروستاتيكية او اشعة ليزر فى تكوين شكل الرمز على الورق . تطبع طابعات الرمز character printer رمزا واحدا بينما تطبع طابعات الاسطر line printers سطرًا كاملا فى نفس الوقت كما تستطيع طابعات الاسطر التى تعمل بأشعة الليزر طباعة مايزيد عن 20000 سطر فى الدقيقة الواحدة .

النهايات الطرفية للنسخ الدائمة، وأنابيب أشعة الكاثود

الوحدات التى سبق شرحها فى القسم ١ - ٢ كوحدات مدخلات ومخرجات معا .

مقبات بطاقات

تعمل هذه الوحدات تحت تحكم مباشر من وحدة التشغيل المركزية، فتتقب المعلومات على هيئة ثقب فى البطاقات .

القرص الصلب

تستطيع مشغلات الاقراص التى تدخل معلومات من اقراص مرنة أو اقراص صلبة ان تسجل كذلك معلومات على الاقراص.

١ - ٤ التخزين المساعد

هى وحدات قادرة على كل من المدخلات والمخرجات، ويمكن ان تخدم كوحدات تخزين مساعد auxiliary storage devices وهى مطلوبة بسبب :

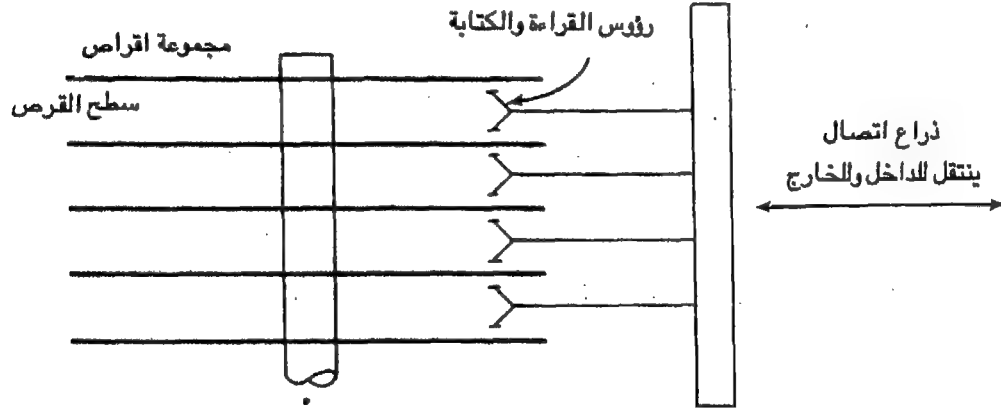
- (١) حتى ذاكرات الكمبيوتر الكبيرة لايمكنها ان تحتفظ بكل المعلومات اللازمة لمجال اعمال صغيرة نسبيا .
- (٢) عند انقطاع التيار عن اجهزة الكمبيوتر.. تعصى كل المعلومات الموجودة فى ذاكرة الكمبيوتر ، فاذا تواجدت كل المعلومات المطلوبة داخل الذاكرة ، فستكون هناك مخاطرة غير مقبولة بحفظها فى الذاكرة .

القرص الصلب

فى الجزء الذى يعد اهم وحدة تخزين مساعد.. يستخدم مشغل الاقراص فى قراءة بيانات من مجموعة اقراص disk pack والكتابة عليها . وتحتوى مجموعة الاقراص على رصعة من اقراص مسطحة من الالومنيوم مغطاة بمادة مغناطيسية موضوعة على محور مركزى (شكل ١ - ٢) . يتحكم ذراع اتصال access arm فى مجموعة من رؤوس القراءة والكتابة read/write heads (رأس واحدة لكل سطح) ، ويمكنه ان ينتقل للداخل والخارج . فى موضع السكون.. يضع ذراع الاتصال الرؤس مواجهة لمجموعة من المسارات tracks الدائرية الموجودة على اسطح مجموعة الاقراص وتسجل البيانات على هذه المسارات . والمسارات مرتبة على هيئة مجموعات تسمى اسطوانات cylinders وهى مجموعات المسارات التى يمكن الاتصال بها فى اى موضع للسكون بآلية الاتصال .

يتراوح متوسط وقت نقل ذراع الاتصال الى الاسطوانة المطلوبة (وقت الاتصال access time) بين ١٦ و ٦٠ (مللى ثانية millisecond) هى واحد من الف من الثانية) . يجب ان يضاف وقت تلخز الدوران rotational delay الى هذا الوقت (الوقت اللازم للوصول للنقطة المطلوبة من المسار تحت الرأس) والوقت (الضئيل) اللازم لنقل البيانات الى او من الذاكرة . ويكون اجمالى وقت الاسترجاع retrieval time حوالى ٢٠ مللى ثانية لاقراص IBM .

لبعض الاقراص (تسمى ثابتة الرأس لكل مسار fixed head-per track) رأس لكل مسار من مسارات مجموعة الاقراص وبهذا يقل وقت الاتصال الى حوالى صفر .



شكل (١ - ٢)

الشريط المغناطيسي

يشبه الشريط المغناطيسي للكمبيوتر - في الاحساس - شريط التسجيل المنزلى ويمكن افه على بكرات او على هيئة كاسيت. تسجل المعلومات مغناطيسيا على طول الشريط. وهناك عيب اساسى فى الشريط كتخزين مساعد، وهو انه يجب ان يقرأ مشغل الشرائط كل المعلومات التى تسبق قطعة المعلومات المطلوبة للحصول على هذه القطعة . وعلى هذا .. فالشريط كفه اذا ما أجرينا المعلومات فى نفس التتابع التى سبق تخزينها على الشريط به فقط (تشغيل تتابعى sequential processing) . يمكن تشغيل المعلومات المخزنة على القرص باى ترتيب نريده (تشغيل عشوائى random processing) .

١- ٥ الذاكرة

تتكون ذاكرة الكمبيوتر من مكونات إلكترونية يمكن ان تخزن كل منها رقم ثنائى واحد (bit) binary digit من المعلومات . وحيث ان (bit) يمكن ان يكون لها القيمة 0 أو 1 فقط .. فتجمع (bit) فى وحدات من ثمانية (عادة) تسمى بايت byte . والبايت الواحد هو جزء من ذاكرة الكمبيوتر يكفى لأن يخزن رمزا character واحداً من المعلومات . وبايت الذاكرة مرقمة . ١ و ٢ و ٣ و ... والعدد الفردى المحدد لتعريف بايت يسمى عنوانا address . ويمكن ان يتحقق الاتصال العشوائى باى عنوان مطلوب فى حوالى نانو ثانية nano second (نانوثانية هى ١ من مليون من الثانية) . تقاس سعة الذاكرة وسعة التخزين المساعد بوحدات كيلوبايت kilobyte (الكيلوبايت هو حوالى ١٠٠٠ بايت) ووحدات ميجابايت megabyte (الميجابايت هو حوالى مليون بايت) . وتسمى الذاكرة التى سعتها حوالى ١٢٨٠٠٠ بايت ذاكرة لها ١٢٨ كيلوبايت بينما تسمى الذاكرة التى سعتها حوالى ٢٠٠٠٠٠٠ بايت ذاكرة لها ٢ ميجابايت . ومجموعات الاقراص النمطية يمكن ان تخزن ما بين ٢٩ الى ١٢٦٠ ميجابايت طبقا لطرازها . يسهم السريان الدائم للمعلومات من وحدات المدخلات او التخزين المساعد الى الذاكرة وتشغيل المعلومات اثناء وجودها فى الذاكرة والنقل النهائى للنتائج الى وحدات المخرجات او التخزين المساعد - فى خاصية دورة مدخلات - تشغيل - مخرجات in-put-process-output cycle لكل تشغيل بيانات الاعمال تقريبا .

١ - ٦ وحدة الحساب والمنطق ALU

وحدة الحساب والمنطق (ALU) هي جزء من وحدة التشغيل المركزية CPU ، التي تقوم بتشغيل الفعل للمعلومات . تستقبل وحدة الحساب والمنطق البيانات التي يجرى عليها التشغيل من الذاكرة . وتنفذ وحدة الحساب والمنطق نوعين أساسيين من تشغيل البيانات .

الحساب : عمليات الجمع والطرح والضرب والقسمة

المنطق : الوظيفة التي تقارن فيها وحدة الحساب والمنطق القيم المخزنة في موقعي ذاكرة مختلفين لتحديد ما إذا كانت القيمتان متساويتين ، فإذا لم يكن الحال كذلك، فأيهما أكبر من الأخرى . ويمثل هذا القرار البسيط أساس كل أنشطة الكمبيوتر المعقدة والتي تبدو ذكية .

١ - ٧ وحدة التحكم CU

تعمل أجهزة الكمبيوتر الحديثة بمفهوم البرنامج المخزن stored program الذي يسمح لنا بتغيير النشاط الذي يؤديه الكمبيوتر (مثل الرواتب أو المخزون) ببساطة، وذلك بإدخال تعليمات متتالية (برنامج) مختلفة في الذاكرة . وتصميم وحدة التحكم لتنفيذ execute البرنامج المخزن على النحو التالي :

(١) أول شيء تفعله وحدة التحكم هو إحضار fetch إحدى التعليمات المخزنة، والتي يكون عنوانها في منطقة تخزين خاصة تسمى مسجل عنوان تعليمات (IAR) Instruction Address Register.

(٢) تقوم وحدة التحكم بعد ذلك بفك شفرة decode هذه التعليمات التي أحضرت من الذاكرة ، أي أنها تحلل التعليمات لتحديد المطلوب عمله (أي عملية operation مطلوب أدائها)، وما هي البيانات المطلوب استخدامها في هذه العملية (ما عناوين الذاكرة للعوامل التي تجرى عليها العملية operation) .

(٣) تستبدل وحدة التحكم عند ذلك العنوان في مسجل عنوان التعليمات بعنوان التعليمات المخزنة التالية .

(٤) أخيراً .. ترسل وحدة التحكم إشارات إلى بقية النظام للتأكد أن العملية المحددة قد نفذت .

أي بالنسبة لعملية جمع ترتب وحدة التحكم للذاكرة بأن ترسل نسخاً من القيم المراد جمعها إلى وحدة الحساب والمنطق . ويتسبب عند ذلك في أن تنفذ وحدة الحساب والمنطق عملية الجمع فعلاً وترسل حاصل الجمع إلى موقع الذاكرة المطلوب . ولاحدى عمليات المدخلات أو المخرجات تنشيط وحدة المدخلات أو المخرجات المحددة والذاكرة متسببة في النقل المناسب للبيانات .

عندما تتم هذه المراحل الأربع لعمل وحدة التحكم ، تعاد الدورة ببساطة مرة أخرى من الخطوة الأولى . (يعاد اعداد مسجل عنوان التعليمات في الخطوة الثالثة).

يستطيع المبرمج أن يغير من هذا التنفيذ المتتابع sequential execution المعتاد، وذلك بوضع تعليمات فرعية -branch in- structions في برنامجه . تغير التعليمات ؛ الفرعية من مسجل عنوان التعليمات بحيث أنه يحتوى على عنوان غير عنوان التعليمات الطبيعية التالية ، ويتسبب ذلك في بدء الكمبيوتر التنفيذ المتتالي الطبيعي عند العنوان الفرعي المحدد .

ونقيس السرعة التي يمكن لأجهزة الكمبيوتر الحديثة أن تنفذ بها التعليمات بعدد ملايين التعليمات المنفذة في الثانية الواحدة (MIPS) millions instructions per second . وتعمل أجهزة الكمبيوتر المتاحة في الأسواق حالياً بسرعات تتراوح من ٢٠ إلى ١١ مليوناً من التعليمات في الثانية الواحدة ومن المتوقع أن تتزايد هذه القيم باستمرار .

١ - ٨ لغات البرمجة

كما تخزن تعليمات البرنامج في ذاكرة الكمبيوتر، وتتخذ في وحدة التشغيل المركزية فإن لها شكل سلسلة من الأرقام الثنائية، ونقول انها معبر عنها بلغة الآلة machine language . الا ان الانسان يجد ان كتابة (وقراءة) البرامج باحدى اللغات مرتفعة المستوى high-level language مثل الكويل مريحة جدا ، ويؤدى الكمبيوتر بنفسه الترجمة اللازمة لبرنامج مكتوب بلغة مرتفعة المستوى (برنامج المصدر source program) الى مايكافته من لغة الآلة (برنامج الهدف object program) وذلك تحت تحكم برنامج خاص يسمى بالترجم Compiler ، وعادة ماتكتب الشركة التى تنتج نظم المكونات المترجمات ، وتحفظ المترجمات فى التخزين المساعد بلغة الآلة .

تمثل عملية الترجمة (من برنامج المصدر المكتوب بالكويل) فى شكل ١ - ٣ . اذا اكتشف المترجم اخطاء فى هيكل (اخطاء تكوينية Syntax errors) برنامج المصدر فانه يتسبب فى طباعة رسائل خطأ تشخيصية diagnostic error messages فى قائمة برنامج المصدر .

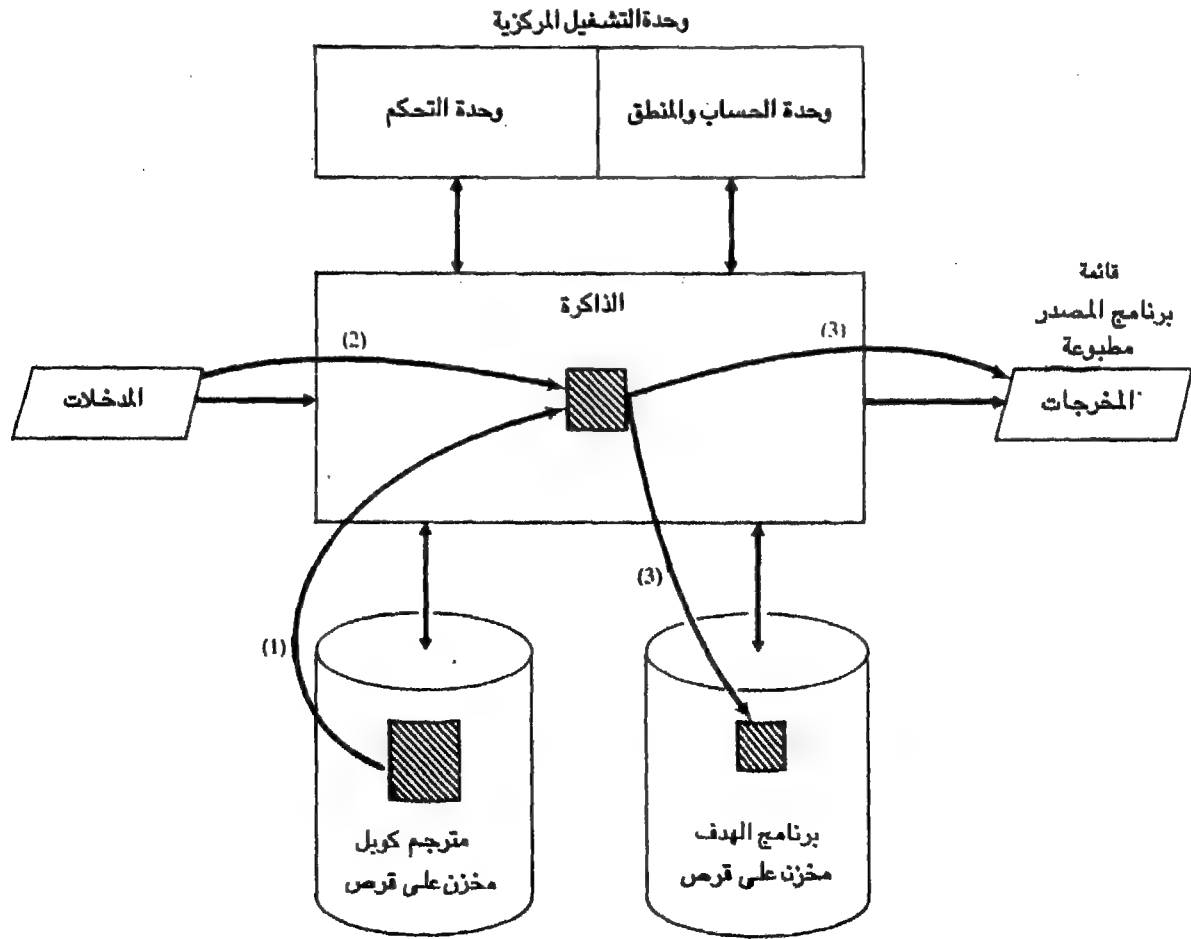
اذا اكتشفت اخطاء تكوينية، فيجب ان يصححها المبرمج ويعيد ترجمة برنامج المصدر المعدل . وإذا لم تكتشف اخطاء فى هذا البرنامج (برنامج المصدر) الجديد ، او برنامج المصدر الاصلى، يجب ان يصحح debugged ؛ أى يجب ان ينفذ بمجموعة من بيانات المدخلات تكون مخرجاتها الصحيحة معروفة مسبقا . وتصمم هذه العملية لاكتشاف اى اخطاء منطقية logic errors موجودة فى البرنامج أى تخطر الآلة بان تضيف (بدلا من الضرب) ساعات العمل ومعدل الاجر فى الساعة إلى حساب اجمالى الاجر . اثناء التصحيح.. يجب ان تعاد ترجمة برنامج المصدر فى كل مرة يحدث فيها تعديل له وذلك لحذف الاخطاء المنطقية . بعد اتمام تصحيح البرنامج.. تحفظ صيغة برنامج الهدف بصفة دائمة فى التخزين المساعد بحيث يمكن قراءته داخل الذاكرة وتنفيذه كلما كانت هناك حاجة لذلك .

١ - ٩ نظم التشغيل

بدلا من جعل المشغل الأدمى operator مسؤولا عن تحميل loading برنامج التنفيذ داخل الذاكرة لتنفيذه.. قام منتجور الكمبيوتر والمتخصصون الآخرون بكتابة برامج تسمى نظم تشغيل operating systems ، التى تسمح للكمبيوتر بعمل هذا النشاط بنفسه وبسرعات الآلة . وتخزن برامج نظام التشغيل هذه بصفة دائمة بلغة الآلة على القرص .

يحتفظ مشغلو الكمبيوتر الأدميون بتحكمهم فى نظام الكمبيوتر عن طريق ادخال معلومات لنظام لتشغيل تذكر له اى برامج اخرى تحمل وتنفذ . وعادة .. مايحدث ذلك بطريقتين . بعض نظم التشغيل مصممة بحيث تقبل عبارات خاصة تسمى لغة تحكم العمل (JCL) job control language من اى وحدة مدخلات . تذكر عبارات لغة تحكم العمل لنظام التشغيل محتويات برامج التطبيقات application programs (مثل : المخزون والرواتب وحسابات المدينين وحسابات الدائنين) التى تحمل داخل الذاكرة . ويصمم بعضها الآخر ليسمح بعلاقة تحاور conversational (او تداخل interactive) بين نظام التشغيل والانسان الموجود امام الشاشة .

يستطيع الانسان كتابة اوامر commands ينفذها نظام التشغيل على الفور . وتذكر الاوامر للنظام البرامج التى يريد المستفيد تحميلها وتنفيذها . ولمارسة البرمجة بالكويل - بالفعل - على جهاز كمبيوتر .. يجب ان تتعلم اولا لغة تحكم العمل او لغة الاوامر الخاصة بنظام تشغيل الجهاز .



- (١) يتم إحضار المترجم من التخزين المساعد إلى داخل الذاكرة
 (٢) ينفذ الكمبيوتر المترجم متسبباً في قراءة وترجمة برنامج المصدر المكتوب بالكويل
 (٣) تتسبب تعليمات المترجم في وضع برنامج الهدف على القرص (لاستخدامه فيما بعد) ، وطباعة قائمة ببرنامج المصدر.

شكل (١ - ٣)

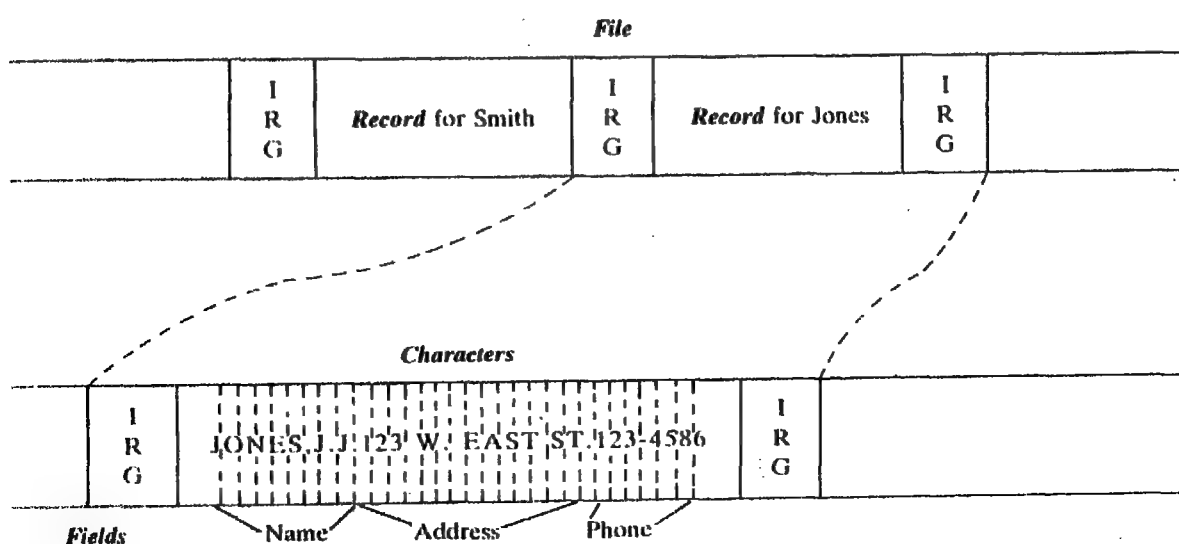
١ - ١٠ الملفات

تحفظ كل من البرامج والبيانات في التخزين المساعد على هيئة ملفات. ويحتوي الملف file على معلومات عن برنامج خاص أو تطبيق خاص ويعرف به «اسم الملف» file name . فمثلاً... يمكن للملف المسمى JAN. PAYROLL ان يحتوي على معلومات عن حسابات المدينين . وتعتمد القواعد الدقيقة لعمل اسماء الملفات على نظام التشغيل الخاص بجهاز الكمبيوتر .

تنقسم الملفات الى سجلات ، يحتوى السجل منها record على معلومات عن شخص معين او شيء معين . إذ، يمكن ان يوجد بالملف المسمى EMPLOYEE. DATA سجل معلومات واحد عن شخص اسمه Smith وسجل معلومات آخر عن شخص اسمه Jones . والسجلات الموجودة على شريط او على قرص تفصل عن بعضها البعض واقعياً بواسطة مناطق فارغة تسمى بفراغات ما بين السجلات interrecord gaps .

وتنقل مشغلات الشرائط والاقراص الخاصة بالسجلات الى ذاكرة الكمبيوتر سجلاً واحداً في نفس الوقت one record at a time . وعلى هذا فعندما يقوم برنامج الكوبل بتشغيل ملف فيجب ان يعمل مع سجل واحد في نفس الوقت . ويجب ان يفكر المبرمج في مصطلحات دورة المدخلات والتشغيل والمخرجات : اى ادخال سجل من الملف وتشغيل السجل واخراج النتائج وتكرار ذلك حتى يتم تشغيل جميع السجلات .

وينقسم السجل الى حقول ، يعد الحقل field الواحد منها قطعة محددة من المعلومات ، مثل الاسم او رقم الحساب ، او اجمالى الراتب او غيرها . ويتكون كل حقل بدوره من مجموعة من البايت byte الفردية (او الرموز الفردية) التى تعطى معناه . وتوضح هرمية الملف والسجل والحقل والرمز فى شكل (١ - ٤) .



شكل (١ - ٤)

تحتفظ معظم نظم التشغيل بعنوان ملف file label لكل ملف على شريط او على قرص . وعنوان الملف هو سجل خاص يحتوى على حقول مثل : اسم الملف الذى يحدد عند انتاج الملف ، وطول الملف وموقع الملف (على القرص) وحجم الملف والمعلومات الخاصة بكلمة المرور التى تؤمن الملف من الاتصال غير المسئول ... الخ . وتسبق عناوين ملف الشريط محتويات الملف مباشرة كما تتبعها مباشرة ، وعادة ما تجمع عناوين ملف القرص مع بعضها فيما يسمى جدول محتويات الحجم volume (VTOC) او دليل directory . ويستطيع نظام التشغيل تحديد موقع اى ملف موجود على مجموعة الاقراص بالبحث عن عنوانه فى الدليل . ويساعد استخدام العناوين فى التأكد من ان البرامج تقوم بتشغيل الملفات الصحيحة .

١ - ١ شفرات البيانات

الوحدة الاساسية للذاكرة والتخزين المساعد هى ، كما نعرف ، الرقم الثنائى او البت . ويجب أن تكتب أى معلومات مخزنة فى هذه الاجزاء من النظام بشفرة coded كسلسلة من الارقام الثنائية . وبهدف عمل الشفرة فمن المقنع تقسيم البيانات كما يلى :

(١) بيانات حرفية عددية alphanumeric أو alphameric يمكن ان تتكون من أى رموز من أى لوحة مفاتيح للمدخلات او أى شاشة مرئية للمخرجات او أى طابع . وتشمل هذه الفئة كل الحروف letter الهجائية والارقام digits من ٠ الى ٩ وكل الرموز الخاصة special characters مثل : (@ , # , \$, % , & , *) و (- و + الفراغ .. الخ) الحقل "John W. Doe" هو حقل حرفى عددى (لاحظ الرمز الخاص ، النقطة) كما ان الحقل "١٧٦-٢٢٨" هو حقل حرفى عددى كذلك (لاحظ الرمز الخاص الشرطة) .

(٢) بيانات حرفية alphabetic تتكون من الحروف الهجائية والفراغ . الحقل "John W Doe" هو حقل حرفى ، وكذلك الحقل "N SPINT ST" هو حقل حرفى ، اما الحقلان "John W. Doe" و "43 N SPRINT ST" فليسا حرفيين .

(٣) بيانات عددية numeric تتكون من الارقام من ٠ الى ٩ مع وجود اشارة سالبة او موجب اختيارية . فى بعض الحالات يمكن ان تحتوى البيانات العددية على علامة عشرية ، الا انه عادة ما يفترض assumed وجود العلامة العشرية ومن أمثلة البيانات العددية مايلى : "1278" ، "53 -" ، "1.28" ، "28 1" ؛ حيث يمثل الرمز - علامة عشرية مفترضة .

تستخدم الشفرات codes التالية فى تمثيل بيانات حرفية عددية ، وحرفية وعددية كاعداد ثنائية مناسبة لتخزينها فى نظام الكمبيوتر .

(١) EBCDIC, DISPLAY (EBCDIC & ASCII) هى شفرة التبادل العشرية المتسعة للشفرة الثنائية Extended American Binary- Coded Decimal Interchange Code ASCII ، هى الشفرة الامريكية النمطية لتبادل المعلومات Standard Code For Information Interchange ، وهما اكثر شفرتين مستخدمتين فى تمثيل البيانات الحرفية عددية . وتمثل كل من الشفرتين كل رمز حرفى عددى (بما فى ذلك الفراغ) برقم ثنائى له ٨ بايت (حيث يتطلب كل رمز بايت واحد من الذاكرة او التخزين المساعد) . والشفرة الكاملة مسرودة فى ملحق ب .

(٢) COMP-3 (packed-decimal) . العشرى المضغوط Packed-decimal هى شفرة تستخدم على نحو صارم مع البيانات العددية فى الذاكرة او التخزين المساعد ولا تستخدم بواسطة أى وحدة مدخلات او مخرجات ، كما انها ليست متاحة على كل اجهزة الكمبيوتر . ويقال فى الكويل - عن اعداد العشرى المضغوط - بانها صورة COMP-3 . وشفرة العشرى المضغوط المميزات التالية عن شفرة EBCDIC ، وشفرة ASCII :

(أ) يمكن لوحد الحساب والمنطق اجراء الحسابات على اعداد العشرى المضغوط مباشرة (اذا دخلت الاعداد الحسابات وهى فى شفرة EBCDIC او شفرة ASCII إذ يجب ان يحولها برنامج التنفيذ الى العشرى المضغوط مؤقتا لاجراء الحسابات ثم يعيد تحويل النتيجة الى شفرة EBCDIC او شفرة ASCII) .

(ب) تشغل اعداد العشرى المضغوط عددا اقل من البايت عن نفس الاعداد المثلة بشفرة EBCDIC او شفرة ASCII .

(٣) COMP (binary-twos-complement) . مكمل الأزواج الثنائية ، وله نفس التطبيق والمميزات التى للعشرى المضغوط ، ويقال فى الكويل عن الاعداد المثلة بمكمل الأزواج الثنائية بانها فى صورة COMP .

وكل كمبيوتر تقريبا يدعم بعض مكملي الأزواج الثنائية ، كما ان بعضها يدعم العشرى المضغوط كذلك . انظر الفصل الخامس لمناقشة وقت استخدام أى من الشفرات السابقة .

الفصل الثانى

عرض سريع للكوبل

Cobol Overview

إحدى سمات الكوبل المهمة هي التوثيق الذاتى self- documenting : يستطيع المبرمج أن يفهم معظم أجزاء برنامج الكوبل بقراءة شفرة البرنامج ببساطة . أفحص برنامج الكوبل الكامل الموجود فى مثال (٢ - ١) الذى تتكرر الإشارة إليه فى هذا الفصل .

مثال ٢ - ١ :

```
00001 IDENTIFICATION DIVISION.
00002
00003 PROGRAM-ID. QUARTER.
00004 AUTHOR. LARRY NEWCOMER.
00005 INSTALLATION. PENN STATE UNIVERSITY--YORK CAMPUS.
00006 DATE-WRITTEN. MAY 1983.
00007 DATE-COMPILED. MAY 9,1983.
00008 SECURITY. THERE ARE NO SECURITY CONSIDERATIONS FOR QUARTER.
00009 *
00010 *OVERVIEW OF PROGRAM QUARTER--
00011 *
00012 * QUARTER READS A FILE CONTAINING SALESPERSON MONTHLY
00013 * SALES FOR THE 3 MONTHS IN A QUARTER ALONG WITH THE
00014 * SALESPERSON'S NAME AND QUARTERLY QUOTA SALES AMOUNT.
00015 * IT PRINTS A REPORT SHOWING SALESPERSON:
00016 *
00017 * NAME QUARTERLY SALES QUOTA
00018 *
00019 * JONES $42,000.98 $40,000.00
00020 * SMITH $59,000.67 $60,000.00
00021 * YOST $47,893.00 $45,000.00
00022 *
00023
00024 ENVIRONMENT DIVISION.
00025
00026 CONFIGURATION SECTION.
00027 SOURCE-COMPUTER. IBM-370.
00028 OBJECT-COMPUTER. IBM-370.
00029
00030 INPUT-OUTPUT SECTION.
00031 FILE-CONTROL.
00032 SELECT SALES-FILE ASSIGN TO SALES.
00033 SELECT QUARTERLY-REPORT ASSIGN TO SALESRPT.
```

```

00034
00035      DATA DIVISION.
00036
00037      FILE SECTION.
00038
00039      FD SALES-FILE
00040          LABEL RECORDS ARE STANDARD
00041          RECORD CONTAINS 80 CHARACTERS
00042      .
00043      01 SALES-RECORD.
00044          05 SALES-RECORD-NAME                PIC X(15).
00045          05 SALES-RECORD-MONTH-1-SALES        PIC S9(4)V99.
00046          05 SALES-RECORD-MONTH-2-SALES        PIC S9(4)V99.
00047          05 SALES-RECORD-MONTH-3-SALES        PIC S9(4)V99.
00048          05 SALES-RECORD-QUOTA                PIC S9(5)V99.
00049          05 FILLER                            PIC X(40).

00050
00051      FD QUARTERLY-REPORT
00052          LABEL RECORDS ARE OMITTED
00053          RECORD CONTAINS 132 CHARACTERS
00054      .
00055      01 QUARTERLY-REPORT-LINE                PIC X(132)..
00056
00057      WORKING-STORAGE SECTION.
00058
00059      01 SWITCHES-AND-TOTALS.
00060          05 SALES-FILE-END                    PIC X.
00061          05 QUARTERLY-TOTAL                    PIC S9(5)V99      COMP-3.
00062      01 WORKING-REPORT-LINE.
00063          05 WORKING-NAME                      PIC X(15).
00064          05 FILLER                            PIC X(5)      VALUE SPACES.
00065          05 WORKING-TOTAL                    PIC $$$,$$$-99.
00066          05 FILLER                            PIC X(5)      VALUE SPACES.
00067          05 WORKING-QUOTA                    PIC $$$,$$$-99.
00068          05 FILLER                            PIC X(87)     VALUE SPACES.
00069
00070      PROCEDURE DIVISION.
00071
00072      010-EXECUTIVE-PARA.
00073          PERFORM 020-INITIALIZE-AND-SET-UP
00074          PERFORM 040-PRINT-REPORT-LINES
00075              UNTIL SALES-FILE-END = "T"
00076          PERFORM 050-TERMINATION-AND-WIND-UP
00077          STOP RUN
00078      .
00079
00080      020-INITIALIZE-AND-SET-UP.
00081          OPEN INPUT SALES-FILE
00082              OUTPUT QUARTERLY-REPORT
00083          MOVE "F" TO SALES-FILE-END
00084          PERFORM 030-READ-SALES-FILE
00085      .
00086
00087      030-READ-SALES-FILE.
00088          READ SALES-FILE
00089              AT END
00090              MOVE "T" TO SALES-FILE-END
00091
00092

```

```

00093      040-PRINT-REPORT-LINES.
00094      MOVE SALES-RECORD-NAME TO WORKING-NAME
00095      COMPUTE QUARTERLY-TOTAL = SALES-RECORD-MONTH-1-SALES
00096      + SALES-RECORD-MONTH-2-SALES
00097      + SALES-RECORD-MONTH-3-SALES
00098      MOVE QUARTERLY-TOTAL TO WORKING-TOTAL
00099      MOVE SALES-RECORD-QUOTA TO WORKING-QUOTA
00100      WRITE QUARTERLY-REPORT-LINE
00101      FROM WORKING-REPORT-LINE
00102      PERFORM 030-READ-SALES-FILE
00103
00104
00105      050-TERMINATION-AND-WIND-UP.
00106      CLOSE SALES-FILE
00107      QUARTERLY-REPORT
00108

```

٢ - ١ أجزاء الكوبل

يحتوى الكوبل دائما على أربعة أجزاء رئيسية DIVISIONS ، وتكون بنفس الترتيب التالى : التعريف IDENTIFICATION ، ثم الأوساط ENVIRONMENT ، ثم البيانات DATA ، ثم الإجراءات PROCEDURE . وفى بداية كل جزء .. توجد عبارة كويل خاصة تسمى عنوان الجزء division header .

مثال ٢ - ٢ :

عناوين الأجزاء فى مثال (٢ - ١) هى :

السطر رقم ١ : IDENTIFICATION DIVISION جزء التعريف .

السطر رقم ٢٤ : ENVIRONMENT DIVISION جزء الأوساط .

السطر رقم ٣٥ : DATA DIVISION جزء البيانات .

السطر رقم ٧٠ : PROCEDURE DIVISION جزء الإجراءات .

يقدم جزء التعريف اسم البرنامج ، ومن الذى كتبه ، ومتى كتب وأين كتب ، ومتى تمت ترجمته ، وماهى احتياطات الأمن (إذا كانت هناك مثل هذه الاحتياطات) التى يجب اتخاذها لتقييد الاتصال بالبرنامج ، وبالملفات التى يقوم بتشغيلها . ويجب أن ينتهى هذا الجزء بمجموعة جمل إنجليزية تعطى عرضاً سريعاً لما يفعله البرنامج (الأسطر من ٩ - ٢٢ موضحة فى مثال (٢ - ١) .

يحتوى جزء الأوساط على معلومات عن الكمبيوتر الذى يترجم عليه برنامج الكويل ، والذى ينفذ عليه البرنامج المكتوب بلغة الآلة ، والنتائج من عملية الترجمة . كما يعطى كذلك اسم كويل لكل ملف يراد تشغيله ، ويحدد وحدة مدخلات أو وحدة مخرجات لكل ملف .

تعرف هذه المعلومات وسط نظم المكونات hardware environment الذى يتم تنفيذ البرنامج فيه .

مثال ٢ - ٢ :

فى مثال (٢ - ١) :

• السطر رقم ٢٧ : SOURCE-COMPUTER. IBM-370 يحدد الكمبيوتر المستخدم فى برنامج المصدر .

• السطر رقم ٢٨ : OBJECT-COMPUTER. IBM-370 يحدد الكمبيوتر المستخدم فى تنفيذ الهدف .

• السطران ٣٢ و ٣٣ : SELECT SALES-FILE ASSIGN TO SALES

SELECT QUARTERLY- REPORT ASSIGN TO SALESRPT

يحددان وحدات المدخلات والمخرجات للملفات .

يعطى جزء البيانات وصفا موجزا لكل ملف يجرى عليه تشغيل، ويعطى كذلك تخطيطا تفصيليا للسجلات الموجودة فى الملف . يوصف كل سجل فى سجل بالنسبة لطوله ونوع بياناته، وكذلك يصف جزء البيانات حقول البيانات التى يستخدمها البرنامج ، ولا تظهر فى سجلات الملفات . مثل هذه الحقول.. تخزن فى مناطق ذاكرة ، تسمى مخزن العمل WORKING-STORAGE، ويجب أن توصف كل حقول البيانات (فى سجلات الملف، أو فى مخزن العمل) فى جزء البيانات .

مثال ٢ - ٤ :

تصف الأسطر من ٣٩ - ٤٩ فى مثال (٢ - ١) ملف المدخلات للمبيعات والسجلات التى يحتويها .

```
FD SALES-FILE
  LABEL RECORDS ARE STANDARD
  RECORD CONTAINS 80 CHARACTERS

01 SALES-RECORD.
  05 SALES-RECORD-NAME          PIC X(15).
  05 SALES-RECORD-MONTH-1-SALES PIC S9(4)V99.
  05 SALES-RECORD-MONTH-2-SALES PIC S9(4)V99.
  05 SALES-RECORD-MONTH-3-SALES PIC S9(4)V99.
  05 SALES-RECORD-QUOTA         PIC S9(5)V99.
  05 FILLER                     PIC X(40).
```

تصف الأسطر من ٥٧ - ٦٨ عناصر بيانات مخزن العمل ، التى لاتكون جزءاً من أى سجلات للملفات .

WORKING-STORAGE SECTION.

```
01 SWITCHES-AND-TOTALS.
  05 SALES-FILE-END          PIC X.
  05 QUARTERLY-TOTAL         PIC S9(5)V99 COMP-3.

01 WORKING-REPORT-LINE.
  05 WORKING-NAME            PIC X(15).
  05 FILLER                  PIC X(5) VALUE SPACES.
  05 WORKING-TOTAL           PIC $$$,$$$,99.
  05 FILLER                  PIC X(5) VALUE SPACES.
  05 WORKING-QUOTA           PIC $$$,$$$,99.
  05 FILLER                  PIC X(87) VALUE SPACES.
```

وفى جزء الإجراءات .. يتم إخطار الكمبيوتر فعلا بالتشغيل المراد أدائه . وتشبه التوجيهات الجمل الإنجليزية بدرجة كبيرة

مثال ٢ - ٥ :

عبارات جزء الإجراءات التالية مأخوذة من مثال ٢ - ١ .

PERFORM 020- INITIALIZE- AND- SET- UP •

(السطر ٧٣) يتسبب فى تنفيذ كل العبارات الموجودة فى المقطع المسمى 020- INITIALIZE- AND- SET- UP ، وهى

الأسطر من ٨٠ - ٨٥ ، وذلك مرة واحدة قبل أن يستمر الكمبيوتر فى تنفيذ العبارة التالية لعبارة :

PERFORM 020- INITIALIZE- AND- SET- UP

(أى السطر رقم ٧٤)

PERFORM 040- PRINT- REPORT- LINES- UP UNTIL SALES- FILE- END= " T " •

(السطران ٧٤ - ٧٥) يتسببان فى أن ينفذ الكمبيوتر العبارات الموجودة فى المقطع المسمى 040- PRINT- REPORT-

LINES . الأسطر من (٩٣ - ١٠٢) وذلك مرات ومرات على التوالى، حتى يحتوى عنصر البيانات المسمى SALES- FILE-

END على القيمة " T " . عندما يحدث ذلك .. ينتقل الكمبيوتر إلى العبارة التالية (أى السطر ٧٦) .

• OPEN INPUT SALES- FILE OUTPUT QUARTERLY- REPORT

(السطران ٨١ - ٨٢) تعد عبارة OPEN الملف للتشغيل، وتحدد ما إذا كان الملف سيستخدم كمدخلات أو مخرجات، ويجب

أن يفتح الملف قبل أن يستخدم فى البرنامج.

• MOVE SALES- RECORD- NAME TO WORKING- NAME

(السطر ٩٤) تنسخ عبارة MOVE محتويات أحد مواقع التخزين (فى هذه الحالة الموقع المحدد لعنصر البيانات المسمى

SALES- RECORD- NAME فى موقع تخزين آخر (فى هذه الحالة الموقع المحدد لعنصر البيانات المسمى WORKING-

NAME).

COMPUTE QUARTERLY-TOTAL = SALES-RECORD-MONTH-1-SALES
+ SALES-RECORD-MONTH-2-SALES
+ SALES-RECORD-MONTH-3-SALES

(الأسطر ٩٥ - ٩٧) تستخدم عبارة COMPUTE فى إجراء حسابات عديدة، تخزن نتيجة الحسابات فى عنصر بيانات

(أى فى موقع الذاكرة المحدد لعنصر البيانات)، الذى يكتب اسمه فى الناحية اليسرى من علامة التساوى.

٢ - ٢ الأقسام والمقاطع

يتكون كل جزء من أجزاء برنامج الكوبل من أقسام SECTIONS، تبدأ فى أسطر خاصة، تسمى عناوين الأقسام sec-

tion headers، ويتكون كل قسم بدوره من مقاطع paragraphs.

مثال ٢ - ٦ :

عناوين الأقسام فى مثال ٢ - ١ هى :

السطر رقم ٢٦ : CONFIGURATION SECTION ، قسم التشكيل .

السطر رقم ٣٠ : INOUT- OUTPUT SECTION ، قسم المدخلات والمخرجات .

السطر رقم ٣٧ : FILE SECTION ، قسم الملفات .

السطر رقم ٥٧ : WORKING- STORAGE SECTION ، قسم مخزن العمل .

تحتوى عناوين الأقسام على اسم للقسم، تتبعه كلمة SECTION ثم نقطة. وتعد أسماء الأقسام في جزء الأوساط وجزء البيانات، جزءاً ثابتاً من لغة الكويل، لا يمكن تغييرها (مثل هذه الكلمات تسمى كلمات محجوزة reserved words) . والأقسام اختيارية في جزء الإجراءات، وعندما تستخدم.. يجب أن تكتب اسمائها بنفس قواعد كتابة كلمات الكويل (مثل الأسماء التي يعرفها المستفيد programmer- defined names) . ولا توجد أى أقسام في جزء التعريف .

يمكن أن تكون المقاطع أجزاء.. في قسم، ويمكن أن تكون قائمة بذاتها داخل الجزء . وفي كل الحالات.. تعرف المقاطع بعنوان paragraph header، الذى يحتوى على اسم المقطع فقط تليه نقطة. أسماء المقاطع في جزء التعريف، وجزء الأوساط، وجزء البيانات هو جزء ثابت من الكويل، ولا يمكن تغييرها . والمقاطع في جزء الإجراءات اختيارية، ويضع المبرمج أسماءها .

مثال ٢ - ٧ :

يتكون جزء التعريف من مقاطع فقط (لا توجد به أقسام). أسماء المقاطع المستخدمة في جزء التعريف في المثال ٢ - ١

هى:

PROGRAM-ID	AUTHOR	INSTALLATION	DATE-WRITTEN	DATE-COMPILED
SECURITY				

أسماء المقاطع جزء ثابت من لغة الكويل.

مثال ٢ - ٨ :

يمكن أن يتكون جزء الإجراءات من برنامج الكويل، أو لا يتكون، من أقسام. ويستخدم مثال (٢ - ١) مقاطع فقط في جزء الإجراءات، والأسماء التي عرف المستفيد لها هى :

010-EXECUTIVE-PARA	020-INITIALIZE-AND-SET-UP	030-READ-SALES-FILE
040-PRINT-REPORT-LINES	050-TERMINATION-AND-WIND-UP	

وكذلك بالرغم من أنه ليس مطلوباً في لغة الكويل أن تبدأ أسماء المقاطع بأرقام إلا أن أسماء كل هذه المقاطع بدأت بأرقام، وهذه طريقة تجعل الوصول إلى مقطع معين في البرنامج أسهل بكثير (بافتراض أن الأرقام تظهر بالترتيب)

استخدام النقطة

تتكون المقاطع في جزء الإجراءات من جمل، مثل اللغة الانجليزية، ويعرف المستفيد الجمل باستخدام نقاط. وبالرغم من أن الكوبل يسمح بوجود نقطة في نهاية كل جملة، إلا أنها فكرة جيدة أن تستخدم النقاط في جزء الإجراءات عند الحاجة الضرورية فقط. لاحظ في جزء الإجراءات من مثال ١ - ٢ أن للنقاط أسطر خاصة. ويوصى بهذا الاستخدام للنقطة (النقطة المرتبة - struced period): نظراً لأنه يجعل النقطة مرئية بوضوح، مع السماح بإدخال عبارات جديدة أمام النقطة، ويساعد ذلك في تقليل الأخطاء المنطقية.

مثال ٢ - ٩ :

يجب استخدام النقطة في الكوبل لإنهاء كل مقطع، كما تكون هناك حاجة لها في بعض الأحيان لفصل عبارة عن عبارة أخرى داخل نفس المقطع.

يستخدم جزء الإجراءات في مثال (١ - ٢) (١) نقطة بعد عنوان الجزء (السطر ٧٠)، و(٢) نقطة بعد عنوان كل مقطع (الاسطر رقم ٧٢ و ٨٠ و ٨٧ و ٩٢ و ١٠٥) و(٣) نقطة مرتبة في نهاية كل مقطع (الأسطر : ٧٨ و ٨٥ و ٩٢ و ١٠٢ و ١٠٨).

مثال ٢ - ١٠ :

يحتوي السطران : (٤٢ ، ٥٤) في مثال (١ - ٢) على نقاط مرتبة. وهي أجزاء من جزء البيانات، قد تتطلب حذف أو إضافة لأسطر كوبل، مع إجراء التغييرات على البرنامج، وتسهل النقطة إجراء مثل هذا التغييرات، إلا أن القاعدة المعتادة هي وضع النقاط، خارج جزء الإجراءات، على نفس السطر الذي تنتهي به.

٣ - ٢ قواعد كتابة برامج كوبل

عادة مايكتب مبرمجو الكوبل برامجهم على صيغة كتابة شفرة coding forms خاصة بالكوبل (شكل ١ - ٢) . وتستخدم الأعمدة من ١ - ٦ من كل سطر في كتابة أرقام متتالية تيسر إعادة ترتيب البرنامج إذا حدث خلط لأسطره بطريقة الخطأ (خلط للبطاقات المثقبة) . تستخدم الأعمدة من العمود رقم ٧٢ إلى العمود ٨٠ من كل سطر في تعريف البرنامج. فيكتب اختصار لاسم البرنامج في هذه الأعمدة. الأرقام المتتالية والتعريف مهمة جداً، عندما يتقرب البرنامج في بطاقات، لأن مجموعة البطاقات يمكن أن تقع. عادة ماتكتب البرامج الآن عن طريق نهاية طرفية، ملحق بها شاشة مرئية ويخزن على قرص، وهذا يقلل الاهتمام بالتتابع والتعريف.

تكتب عبارات الكوبل نفسها في الأعمدة من العمود الثامن وحتى العمود الثاني والسبعين. تمثل الأعمدة من (٨ - ١١) المنطقة ٨، وتمثل الأعمدة من العمود (١٢ - ٧٢) المنطقة B. والعناصر التالية يجب أن تبدأ كتابتها في المنطقة A.

- عناوين الأجزاء (أسماء الأجزاء) .
- عناوين الأقسام (أسماء الأقسام) .
- عناوين المقاطع (أسماء المقاطع) .

وداخل جزء البيانات

- عناوين محتويات وصف الملف (يدأ بالأحرف FD ، كما في الأسطر ٢٩ - ٥١ من مثال (١ - ٢) .

- عناوين محتويات وصف السجل (عناصر البيانات التي لها رقم مستوي 01) ، كما في الأسطر رقم ٤٢ ، ٥٥ ، ٥٩ ، ٦٢ في

مثال ٢ - ١١ . يجب أن تبدأ كتابة جميع العناصر الأخرى في المنطقة B . وبالرغم من أن الكوبل يسمح بمحتويات المنطقة B ، لأن تبدأ من العمود رقم ١٢ ... إلا أنه من المهم عمل ترحيل في البرمجة المرتبة داخل المنطقة B لتوضيح العلاقات بين الأسطر.

SYSTEM		PUNCHING INSTRUCTIONS		PAGE	OF
PROGRAM		GRAPHIC	PUNCH	CARD FORM #	
PROGRAMMER		DATE			
SEQUENCE	1	A	B	IDENTIFICATION	
1	1	1	1	1	1
2	2	2	2	2	2
3	3	3	3	3	3
4	4	4	4	4	4
5	5	5	5	5	5
6	6	6	6	6	6
7	7	7	7	7	7
8	8	8	8	8	8
9	9	9	9	9	9
10	10	10	10	10	10
11	11	11	11	11	11
12	12	12	12	12	12
13	13	13	13	13	13
14	14	14	14	14	14
15	15	15	15	15	15
16	16	16	16	16	16
17	17	17	17	17	17
18	18	18	18	18	18
19	19	19	19	19	19
20	20	20	20	20	20
21	21	21	21	21	21
22	22	22	22	22	22
23	23	23	23	23	23
24	24	24	24	24	24
25	25	25	25	25	25
26	26	26	26	26	26
27	27	27	27	27	27
28	28	28	28	28	28
29	29	29	29	29	29
30	30	30	30	30	30
31	31	31	31	31	31
32	32	32	32	32	32
33	33	33	33	33	33
34	34	34	34	34	34
35	35	35	35	35	35
36	36	36	36	36	36
37	37	37	37	37	37
38	38	38	38	38	38
39	39	39	39	39	39
40	40	40	40	40	40
41	41	41	41	41	41
42	42	42	42	42	42
43	43	43	43	43	43
44	44	44	44	44	44
45	45	45	45	45	45
46	46	46	46	46	46
47	47	47	47	47	47
48	48	48	48	48	48
49	49	49	49	49	49
50	50	50	50	50	50
51	51	51	51	51	51
52	52	52	52	52	52
53	53	53	53	53	53
54	54	54	54	54	54
55	55	55	55	55	55
56	56	56	56	56	56
57	57	57	57	57	57
58	58	58	58	58	58
59	59	59	59	59	59
60	60	60	60	60	60
61	61	61	61	61	61
62	62	62	62	62	62
63	63	63	63	63	63
64	64	64	64	64	64
65	65	65	65	65	65
66	66	66	66	66	66
67	67	67	67	67	67
68	68	68	68	68	68
69	69	69	69	69	69
70	70	70	70	70	70
71	71	71	71	71	71
72	72	72	72	72	72
73	73	73	73	73	73
74	74	74	74	74	74
75	75	75	75	75	75
76	76	76	76	76	76
77	77	77	77	77	77
78	78	78	78	78	78
79	79	79	79	79	79
80	80	80	80	80	80
81	81	81	81	81	81
82	82	82	82	82	82
83	83	83	83	83	83
84	84	84	84	84	84
85	85	85	85	85	85
86	86	86	86	86	86
87	87	87	87	87	87
88	88	88	88	88	88
89	89	89	89	89	89
90	90	90	90	90	90
91	91	91	91	91	91
92	92	92	92	92	92
93	93	93	93	93	93
94	94	94	94	94	94
95	95	95	95	95	95
96	96	96	96	96	96
97	97	97	97	97	97
98	98	98	98	98	98
99	99	99	99	99	99
100	100	100	100	100	100

شكل (٢ - ١)

مثال ٢ - ١١ :

فى عبارات جزء الإجراءات التالية من مثال (٢ - ١) تكون الأسطر التى تستمر فيها العبارة - التى بدأت فى السطر السابق - مرحلة تحت السطر الأصيل .
الأسطر من ٨٨ - ٩٠ :

```
READ SALES-FILE
  AT END
    MOVE "T" TO SALES-FILE-END
```

السطران ١٠٠ - ١٠١ :

```
WRITE QUARTERLY-REPORT-LINE
  FROM WORKING-REPORT-LINE
```

استخدامات خاصة للعمود السابع

يفترض الكوبل أن العبارة تستمر ، سطرا بعد سطر، حتى تظهر نقطة أو عبارة أخرى. وهناك حاجة إلى قواعد خاصة للاستمرارية عندما ينتهى السطر فى منتصف الكلمة، أو فى منتصف سلسلة موضوعة بين علامتى تنصيص quoted string فقط (القسم التاسع عشر من الفصل الخامس) :

(١) إذا انتهى السطر السابق فى منتصف الكلمة ... اكتب شرطة - « فى العمود السابع من السطر التالى، واستمر فى كتابة بقية الكلمة فى المنطقة B من السطر التالى .

(٢) إذا انتهى السطر السابق فى منتصف سلسلة موضوعة بين علامتى تنصيص ... اكتب - « فى العمود السابع من السطر التالى، واكتب علامة تنصيص فى أى مكان فى المنطقة B من السطر التالى، واستمر بعدها فى كتابة بقية السلسلة.

مثال ٢ - ١٢ :

```
MOVE "THANK YOU FOR YOUR
      "PROMPT PAYMENT" TO
      CUSTOMER-MESSAGE
col. 7   col. 12   col. 16
```

بينما يعرض استمرار مناسب، فما سبق ذكره ينتج عنه فراغات عديدة غير مطلوبة : لأن أول سطر من السلسلة الموضوعة بين علامتى تنصيص يفترض استمراره حتى العمود ٧٢ (ليس أكثر من ذلك) .

مثال ٢ - ١٣ :

ما الخطأ فى الاستمرار التالى :

```
MOVE "OVER
      "PAYMENT" TO NEGATIVE-BALANCE-MESSAGE
col. 7   col. 12   col. 16
```

(أ) كان يمكن تجنب الاستمرار فى المكان الأول

MOVE "OVERPAYMENT"
TO NEGATIVE-BALANCE-MESSAGE

(ب) نظراً لأن السطر الأول يمتد على طول الطريق حتى العمود ٧٢... فيجب أن تظهر الرسالة فعلاً على النحو التالى:

"OVER

PAYMENT"

الشيء غير المطلوب

عندما تكتب نجمة (*) فى العمود السابع.. تعامل بقية السطر كتعليق comment كويل. وهذا يعنى أنه بالرغم من أن المترجم يطبع السطر فى قائمة برنامج المصدر، إلا أنه لا يترجمه إلى برنامج الهدف. وتستخدم التعليقات لتوضيح مايفعله البرنامج وكيفية أدائه لذلك، وهى موجهة إلى مبرمجى الكويل الآخرين، الذين يحتاجون إلى إجراء تغييرات على البرنامج. تذكر: بما أن الكويل لغة توثيق ذاتي.. فإن التعليقات الزائدة تستخدم فقط عندما تكون هناك حاجة حقيقية لتوضيحات أكثر.

مثال ٢ - ١٤ :

هناك تعليقات موضحة فى الأسطر من ٩ - ٢٢ من مثال (٢ - ١) .

يستخدم العمود السابع كذلك فى إدخال اسطر تصحيح debugging lines فى برنامج الكويل، وهى ببساطة عبارات كويل مع كتابة D فى العمود السابع . كما أسماها.. فهى تستخدم فى المساعدة فى تحديد وتصحيح الأخطاء فى البرنامج (انظر القسم الثانى من الفصل الرابع) .

الأسطر الفارغة

يمكن أن تظهر اسطر فارغة فى برنامج الكويل، وتستخدم لتسهيل قراءة برنامج المصدر (تجعل الاسطر رقم ٢ ، ٢٣ ، ٢٩... و ١٠٤ مثال ٢ - ١ أسهل فى قراءته).

يفضل بعض المبرمجين أسطر تعليقات فارغة blank comment lines (أسطر موضوعة نجمة *) فقط فى عمودها السابع) لأن المترجم يستطيع أن يشغل سطر التعليق أسرع من تشغيله السطر الفارغ كله، إلا أن الميزة صغيرة جداً، وسوف نستخدم أسطر فارغة كلية.

٢ - ٤ الاسماء التى يعرفها المبرمج والكلمات المحجوزة

تتشكل الكلمات الفردية التى يتكون منها برنامج الكويل طبقاً للقواعد التالية :

- (١) لايزيد طولها عن ٣٠ رمزا.
- (٢) الرموز المسموح بها هى الحروف الهجائية من A الى Z ، والارقام من ٠ الى ٩ والشرطة .
- (٣) يجب ألا تبدأ الكلمة أو تنتهى بشرطة .
- (٤) يجب ألا توجد فراغات داخل الكلمة .

تعتبر بعض الكلمات فى برنامج الكويل خاصة بلغة الكويل، وتخدم غرضاً واحداً فقط سبق تعريفه . هذه الكلمات المحجوزة reserved words تشمل : IDENTIFICATION و DIVISION و DATE-WRITTEN و INPUT-OUTPUT و SECTION و WORKING-STORAGE و MOVE و OPEN و COMPUTE، ويوجد بملحق A قائمة كاملة بالكلمات المحجوزة . يجب ألا يكرر هجاء أى كلمة من الاسماء التى يعرفها المبرمج -والتي تعد طبقاً للقواعد سالفة الذكر - أو الكلمات المحجوزة . ومن الكلمات التى يعرفها المبرمج مايلى :

• أسماء الملفات المستخدمة في عبارات SELECT ووصف الملف FD (انظر الاسطر رقم 32, 33, 39, 51) من مثال ١ - ٢

• أسماء السجلات وأسماء الحقول داخل السجلات في جزء البيانات (انظر الاسطر من رقم 43 الى رقم 49 ورقم 55 ومن رقم 59 الى رقم 68) .

• أسماء المقاطع والأقسام في جزء الاجراءات (انظر الاسطر رقم 72, 80, 87, 93, 105) .

مثال ٢ - ١٥ :

(١) قارن أزواج الاسماء التي يعرفها المبرمج التالية :

- YTDAS or YEAR-TO-DATE-AVERAGE-SALES
- GPA or CUMULATIVE-GRADE-POINT-AVERAGE
- EXTRA or YEAR-END-BONUS-PAY

عندما يكون لديك شك.. اعد الهجاء ؛ فمن الصعب ان تقع في الخطأ عندما تصف وصفا دقيقا .

(ب) الاسماء التي يعرفها المستفيد التالية غير صحيحة او غير مرغوب فيها .

• RECORD (غير صحيح : لاستخدامه كلمة محجوزة)

• STUDENT- BODY- CUMULATIVE- GRADE- POINT- AVERAGE

(غير صحيح : لانه اكثر من ٣٠ رمز)

• -OUT- OF- STOCK (غير صحيح لانه بدء بالشرطة)

• N (غير مرغوب فيه : لانه لا يصف شيئا)

• TOTAL- \$ SALES (غير صحيح : لانه يحتوى على رمز ممنوع استخدامه، وهو علامة الدولار)

٢ - ٥ البعد

عند هذه النقطة .. نكون قد خدشنا بالكاد سطح برمجة الكوبل . الا أن أى لغة برمجة يحسن تعلمها باستخدامها . ويخدم برنامجا الكوبل التاليان، كتوجيه في حل تعاريف البرمجة المعطاة في نهاية هذا الفصل . ولاختبار البرامج التي تكتبها فعلا.. عليك باستشارة أستاذك او مبرمج خبير بالنسبة لعبارات تحكم العمل او عبارات لغة الأوامر اللازمة لجهاز الكمبيوتر المتاح لك.

مثال ٢ - ١٦ :

برنامج كوبل كامل ؛ حيث إن كل التعليقات مقدمة كجزء من البرنامج (باستخدام أسطر تعليق) . تلى قائمة برنامج المصدر الفعلية هذه عينة للمخرجات .

```

00001      IDENTIFICATION DIVISION.
00002
00003      PROGRAM-ID. PHONELST.
00004
00005      AUTHOR.  LARRY NEWCOMER.
00006      INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.
00007
00008      DATE-WRITTEN.  MAY 1983.
00009      DATE-COMPILED.  MAY 9,1983.
00011      *      PHONELST PRODUCES A PRINTED LISTING OF ALL EMPLOYEE
00012      *      NAMES AND PHONE NUMBERS.  NAMES AND PHONE NUMBERS ARE
00013      *      INPUT FROM A CARD FILE WHICH IS MAINTAINED IN ALPHABETICAL
00014      *      ORDER BY EMPLOYEE NAME.  THE ONLY OUTPUT IS A PRINTED
00015      *      COPY OF NAMES AND PHONE NUMBERS -- SINGLE SPACED, WITH
00016      *      NO HEADINGS, PAGE NUMBERS, ETC. (KEEP IT SIMPLE)

00018      ENVIRONMENT DIVISION.

00020      CONFIGURATION SECTION.

00022      *      THE CONFIGURATION SECTION DESCRIBES THE COMPUTER SYSTEM
00023      *      TO BE USED TO COMPILE THE COBOL PROGRAM
00024      *      (SOURCE-COMPUTER), AND THE COMPUTER SYSTEM TO BE USED TO
00025      *      EXECUTE THE RESULTING OBJECT PROGRAM (OBJECT-COMPUTER).
00026
00027      SOURCE-COMPUTER.  IBM-370.
00028      OBJECT-COMPUTER.  IBM-370.

00030      INPUT-OUTPUT SECTION.

00032      FILE-CONTROL.

00034      *      THE FILE-CONTROL PARAGRAPH OF THE INPUT-OUTPUT SECTION
00035      *      BEGINS TO DEFINE THE FILES TO BE PROCESSED BY THIS PROGRAM:
00036
00037      SELECT CARD-FILE          ASSIGN TO CARDS.
00038      SELECT PRINT-FILE        ASSIGN TO PRINTER.
00039
00040      *      THESE STATEMENTS GIVE COBOL NAMES TO THE FILES TO BE
00041      *      PROCESSED.  IN THIS CASE THE INPUT FILE IS KNOWN AS
00042      *      CARD-FILE AND THE OUTPUT FILE IS KNOWN AS PRINT-FILE.
00043
00044      *      THE SELECT STATEMENTS ALSO ASSIGN EACH FILE TO A JOB
00045      *      CONTROL LANGUAGE STATEMENT WHICH WILL PROVIDE FURTHER
00046      *      INFORMATION REGARDING THE ACTUAL I/O DEVICE TO BE USED.
00047      *      CONSULT YOUR INSTRUCTOR OR AN EXPERIENCED PROGRAMMER
00048      *      REGARDING WHAT JOB CONTROL LANGUAGE IS NEEDED TO RUN
00049      *      A COBOL PROGRAM ON YOUR SYSTEM.

00051      DATA DIVISION.

00053      *      THE DATA DIVISION DESCRIBES ALL DATA ITEMS TO BE
00054      *      PROCESSED BY THIS PROGRAM.
00055
00056      FILE SECTION.

00058      *      THE FILE SECTION PROVIDES ADDITIONAL INFORMATION FOR
00059      *      EACH FILE NAMED IN A SELECT STATEMENT.  NOTICE THAT
00060      *      THE SELECT FILE NAME AND THE "FD" FILE NAME ARE THE
00061      *      SAME.  THERE MUST BE A FILE DESCRIPTION ("FD") FOR
00062      *      EACH FILE.
00063

```

```

00064      FD  CARD-FILE
00065      RECORD CONTAINS 80 CHARACTERS
00066      LABEL RECORDS ARE OMITTED
00067
00068      *      THE "FD" INDICATES THE NUMBER OF CHARACTERS IN A RECORD
00069      *      AND WHETHER OR NOT THE FILE HAS FILE LABELS.  FOR A
00070      *      STANDARD SIZE CARD, THE LENGTH IS 80 COLUMNS.
00071      *      FILE LABELS ARE NOT PERMITTED FOR CARD FILES.
00072
00073      *      THE REST OF THE "FD" DESCRIBES THE RECORD CONTENTS.
00074      *      "PIC X(80)" MEANS EACH RECORD IS 80 ALPHANUMERIC
00075      *      CHARACTERS.
00076
00077
00078      01  NAME-PHONE-INPUT          PIC X(80).

00080      *      THE "FD" FOR A PRINT FILE MUST INDICATE NO FILE LABELS.
00081      *      MOST PRINTERS PRINT UP TO 132 CHARACTER LINES.
00082
00083      FD  PRINT-FILE
00084      RECORD CONTAINS 132 CHARACTERS
00085      LABEL RECORDS ARE OMITTED
00086
00087
00088      *      A "RECORD" FOR A PRINT FILE IS A PRINTED LINE.  IN
00089      *      THIS CASE EACH LINE CONSISTS OF 132 ALPHANUMERIC
00090      *      CHARACTERS ("PIC X(132)").
00091
00092      01  PRINT-LINE                PIC X(132).

00094      WORKING-STORAGE SECTION.

00096      *      WORKING-STORAGE AREAS ARE NOT PART OF FILE RECORDS.
00097      *      HERE WE DEFINE A PROGRAM SWITCH WHICH CAN HOLD 3
00098      *      ALPHANUMERIC CHARACTERS ("PIC X(3)").  THE SWITCH
00099      *      CONTAINS THE VALUE "NO " AS LONG AS THERE ARE MORE
00100      *      INPUT RECORDS TO BE PROCESSED.  WHEN THERE ARE NO
00101      *      MORE INPUT RECORDS LEFT, THE SWITCH IS SET TO "YES".
00102
00103      01  END-OF-CARDS-SWITCH        PIC X(3).

00105      *      THIS AREA HOLDS AN INPUT CARD WITH AN EMPLOYEE'S
00106      *      NAME AND PHONE NUMBER -- NAME IS 20 ALPHANUMERIC
00107      *      CHARACTERS; PHONE IS 8 ALPHANUMERIC CHARACTERS.
00108      *      "FILLER" IS USED TO DEFINE ANY PART OF A RECORD WHICH
00109      *      WILL NOT BE PROCESSED BY THE PROGRAM.
00110
00111      01  NAME-PHONE-CARD.
00112          05  EMPLOYEE-NAME          PIC X(20).
00113          05  EMPLOYEE-PHONE-NUMBER  PIC X(8).
00114          05  FILLER                 PIC X(52).

00116      *      THIS AREA IS WHERE WE PUT TOGETHER A LINE TO BE PRINTED.
00117      *      NOTICE THERE ARE SOME AREAS BETWEEN DATA ITEMS WHICH
00118      *      HAVE BEEN NAMED "FILLER" AND SET TO BLANK SPACES
00119      *      WITH "VALUE SPACES".  THESE AREAS SEPARATE FIELDS
00120      *      ON THE PRINTED LINE.
00121
00122      01  NAME-PHONE-LINE.
00123          05  PRINT-NAME              PIC X(20).
00124          05  FILLER                  PIC X(5)      VALUE SPACES.
00125          05  PRINT-PHONE-NUMBER     PIC X(8).
00126          05  FILLER                  PIC X(99)     VALUE SPACES.

```

```

00128      PROCEDURE DIVISION.

00130      *          THE PROCEDURE DIVISION CONTAINS THE INSTRUCTIONS
00131      *          WHICH TELL THE COMPUTER WHAT TO DO.
00132
00133      000-PRODUCE-PHONE-LISTING.

00135          MOVE "NO " TO END-OF-CARDS-SWITCH
00136
00137      *          SETS END-OF-CARDS-SWITCH TO "NO".
00138
00139      *          THE OPEN STATEMENT MUST BE USED BEFORE A FILE CAN
00140      *          BE PROCESSED IN ANY WAY. IT TELLS WHETHER THE FILE
00141      *          IS TO BE USED FOR INPUT OR OUTPUT.
00142
00143      OPEN      INPUT      CARD-FILE
00144              OUTPUT     PRINT-FILE
00145
00146      PERFORM 100-INPUT-A-RECORD
00147
00148      *          THE PERFORM STATEMENT CAUSES ALL STATEMENTS
00149      *          IN THE PARAGRAPH NAMED "100-INPUT-A-RECORD" TO BE
00150      *          EXECUTED. NOTE THAT THIS CAUSES THE FIRST RECORD
00151      *          OF THE FILE TO BE INPUT. THE "PRODUCE-NAME-PHONE-
00152      *          LISTING" PARAGRAPH EXECUTED NEXT BEGINS BY
00153      *          PROCESSING THIS FIRST RECORD. WHEN IT IS DONE
00154      *          PROCESSING THE RECORD, IT THEN PERFORMS THE
00155      *          100-INPUT-A-RECORD PARAGRAPH TO OBTAIN THE NEXT
00156      *          RECORD IN THE FILE.
00157
00158
00159
00160      PERFORM 200-PRODUCE-NAME-PHONE-LIST
00161      UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00162
00163      *          THIS VERSION OF PERFORM CAUSES THE ENTIRE
00164      *          PARAGRAPH NAMED "200-PRODUCE-NAME-PHONE-LIST"
00165      *          TO BE EXECUTED REPEATEDLY. THE REPEATED
00166      *          INVOKING OF THE PARAGRAPH CONTINUES UNTIL
00167      *          END-OF-CARDS-SWITCH CONTAINS THE VALUE "YES".
00168
00169
00170      CLOSE  CARD-FILE
00171           PRINT-FILE
00172
00173      *          THE CLOSE STATEMENT MUST BE USED WHEN THE PROGRAM
00174      *          IS DONE PROCESSING A FILE. IT "DISCONNECTS" THE
00175      *          FILE FROM THE COBOL PROGRAM.
00176
00177      STOP RUN
00178
00179      *          THE STOP STATEMENT TELLS THE COMPUTER TO STOP
00180      *          EXECUTING INSTRUCTIONS IN THIS PROGRAM.
00181
00182
00184      100-INPUT-A-RECORD.
00185
00186      *          THIS PARAGRAPH INPUTS A CARD. THE READ
00187      *          STATEMENT INPUTS THE NEXT CARD IN THE FILE AND
00188      *          MOVES THE CONTENTS OF THE CARD INTO THE
00189      *          WORKING-STORAGE AREA NAMED "NAME-PHONE-CARD".
00190      *          IF THERE ARE NO MORE CARDS TO BE READ

```



```

00191      *      ("AT END"), END-OF-CARDS-SWITCH IS SET
00192      *      TO "YES". THIS CAUSES THE CLOSE AND STOP
00193      *      RUN STATEMENTS TO BE EXECUTED.
00194
00195      READ CARD-FILE RECORD
00196      INTO NAME-PHONE-CARD
00197      AT END
00198      MOVE "YES" TO END-OF-CARDS-SWITCH
00199

00201      200-PRODUCE-NAME-PHONE-LIST.
00202
00203      *      THIS PARAGRAPH MOVES THE DATA FROM NAME-PHONE-CARD
00204      *      TO THE AREA FROM WHICH A LINE OF THE REPORT WILL BE
00205      *      PRINTED. THE "WRITE" STATEMENT PRINTS A LINE WHOSE
00206      *      CONTENTS ARE IN "NAME-PHONE-LINE". HAVING PROCESSED
00207      *      ONE EMPLOYEE'S INFORMATION, "PERFORM 100-INPUT-A-RECORD"
00208      *      CAUSES THE READ STATEMENT IN PARAGRAPH 100-INPUT-A-RECOI
00209      *      TO BE EXECUTED, BRINGING THE NEXT RECORD INTO MEMORY
00210      *      FOR PROCESSING.
00211
00212      MOVE EMPLOYEE-NAME          TO PRINT-NAME
00213      MOVE EMPLOYEE-PHONE-NUMBER  TO PRINT-PHONE-NUMBER
00214
00215      WRITE PRINT-LINE
00216      FROM NAME-PHONE-LINE
00217
00218      PERFORM 100-INPUT-A-RECORD
00219

ABEL FRED      123-4567
BAKER SUE      111-2222
CHARLIE SUE    453-7822
PERELMAN BARNEY UNLISTED
PERELMAN MIKE  UNKNOWN
RODGERS BOB    234-5678
RODGERS BOBB   345-6789
RODGERS LEE    456-7890
FOLKERS DICK   567-8901
FOLKERS RUTH   678-9012
SUE PEGGY      231-7856
ZOTZ ZAPPA     999-9999

```

مثال ٢ - ١٧

يشبه مثال ٢ - ١٦ ، وتعلق على السمات الجديدة فقط .

```

00001      IDENTIFICATION DIVISION.
00002
00003      PROGRAM-ID. TIMELIST.
00004
00005      AUTHOR.  LARRY NEWCOMER.
00006      INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.
00007
00008      DATE-WRITTEN.  MAY 1983.
00009      DATE-COMPILED. MAY 9,1983.
00011      *   TIMELIST PRINTS ONE LINE FOR EACH EMPLOYEE SHOWING:
00012      *   NUMBER OF REGULAR HOURS WORKED, NUMBER OF OVERTIME
00013      *   HOURS WORKED, AND TOTAL HOURS WORKED. AT THE END OF THE
00014      *   REPORT, IT ALSO PRINTS A COUNT OF THE NUMBER OF
00015      *   EMPLOYEES PROCESSED. INPUT IS FROM A DECK OF TIME
00016      *   CARDS, KEPT IN SEQUENCE BY EMPLOYEE NAME.

00018      ENVIRONMENT DIVISION.

00020      CONFIGURATION SECTION.

00022      SOURCE-COMPUTER. IBM-370.
00023      OBJECT-COMPUTER. IBM-370.

00025      INPUT-OUTPUT SECTION.

00027      FILE-CONTROL.

00029          SELECT CARD-FILE          ASSIGN TO CARDS.
00030          SELECT PRINT-FILE         ASSIGN TO PRINTER.

00032      DATA DIVISION.

00034      FILE SECTION.

00036      FD  CARD-FILE
00037          RECORD CONTAINS 80 CHARACTERS
00038          LABEL RECORDS ARE OMITTED
00039          .
00040
00041      01  TIME-CARD-INPUT          PIC X(80).

00043      FD  PRINT-FILE
00044          RECORD CONTAINS 132 CHARACTERS
00045          LABEL RECORDS ARE OMITTED
00046          .
00047
00048      01  PRINT-LINE              PIC X(132).

00050      WORKING-STORAGE SECTION.

00052      *   OBSERVE THAT THERE ARE SEVERAL NUMERIC FIELDS DEFINED

00053      *   IN WORKING-STORAGE USING "PIC 99" OR "PIC 999". EACH
00054      *   "9" IN THE DESCRIPTION OF A NUMERIC ITEM REPRESENTS
00055      *   ONE DECIMAL DIGIT POSITION. THUS "PIC 99" DESCRIBES
00056      *   A DATA ITEM WHICH CAN HOLD A 2-DIGIT NUMBER.

00057
00058      01  END-OF-CARDS-SWITCH      PIC X(3).

00060      01  WORKING-TIME-CARD.

00061
00062      *   HOLDS INFORMATION INPUT FROM THE TIME CARD FILE
    
```

```

00063
00064      05 WORKING-NAME          PIC X(20).
00065      05 WORKING-REGULAR-HOURS PIC 99.
00066      05 WORKING-OVERTIME-HOURS PIC 99.
00067      05 FILLER                PIC X(56).

00069      01 TIME-LISTING-LINE.
00070
00071      * AREA USED TO CONSTRUCT A LINE TO BE PRINTED ON THE REPORT
00072
00073      05 LISTING-NAME          PIC X(20).
00074      05 FILLER                PIC X(5)      VALUE SPACES.
00075      05 LISTING-REGULAR-HOURS PIC 99.
00076      05 FILLER                PIC X(5)      VALUE SPACES.
00077      05 LISTING-OVERTIME-HOURS PIC 99.
00078      05 FILLER                PIC X(5)      VALUE SPACES.
00079      05 LISTING-TOTAL-HOURS  PIC 999.
00080      05 FILLER                PIC X(90)     VALUE SPACES.

00082      01 EMPLOYEE-TOTAL-LINE.
00083
00084      * AREA USED TO CONSTRUCT THE LINE TO BE PRINTED AT END OF
00085      * THE REPORT. CONTAINS COUNTER FOR NUMBER OF EMPLOYEES
00086      * PROCESSED.
00087
00088      05 FILLER                PIC X(25)
00089      VALUE "NUMBER OF EMPLOYEES IS".
00090      05 NUMBER-OF-EMPLOYEES  PIC 999.
00091      05 FILLER                PIC X(104)    VALUE SPACES.

00093      PROCEDURE DIVISION.

00095      000-PRODUCE-HOURS-REPORT.
00096
00097      * PROGRAM SWITCHES AND COUNTERS MUST BE INITIALIZED
00098      * AT THE START OF PROGRAM EXECUTION. THE SWITCH IS
00099      * SET TO "NO " WHILE THE COUNTER FOR NUMBER OF
00100      * EMPLOYEES IS INITIALLY SET TO ZERO. ONE WILL BE
00101      * ADDED EACH TIME AN EMPLOYEE IS PROCESSED (SEE
00102      * 200-PRODUCE-TIME-LISTING PARAGRAPH).
00103
00104      MOVE "NO " TO END-OF-CARDS-SWITCH
00105      MOVE ZERO TO NUMBER-OF-EMPLOYEES
00106
00107      OPEN      INPUT          CARD-FILE
00108               OUTPUT         PRINT-FILE
00109
00110      PERFORM 100-INPUT-A-RECORD
00111
00112      PERFORM 200-PRODUCE-TIME-LISTING
00113      UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00114
00115      * AFTER ALL INPUT CARDS HAVE BEEN PROCESSED (I.E.,
00116      * WHEN END-OF-CARDS-SWITCH HAS BEEN CHANGED TO
00117      * "YES"), A LINE CONTAINING A COUNT OF ALL
00118      * EMPLOYEES PROCESSED MUST BE PRINTED.

00119      * "WRITE PRINT-LINE FROM EMPLOYEE-TOTAL-LINE" PRINTS
00120      * THE CONTENTS OF EMPLOYEE-TOTAL-LINE.
00121
00122      WRITE PRINT-LINE
00123      FROM EMPLOYEE-TOTAL-LINE

```

```

00124
00125      CLOSE      CARD-FILE
00126                PRINT-FILE
00127      STOP RUN
00128      .

00130      100-INPUT-A-RECORD.
00131
00132      READ CARD-FILE RECORD
00133      INTO WORKING-TIME-CARD
00134      AT END
00135      MOVE "YES" TO END-OF-CARDS-SWITCH
00136      .

00138      200-PRODUCE-TIME-LISTING.
00139
00140      *          INCREMENT COUNTER EACH TIME AN EMPLOYEE IS PROCESSED
00141
00142      ADD 1 TO NUMBER-OF-EMPLOYEES
00143
00144      *          MOVE DATA FROM INPUT CARD TO AREA WHERE LINE IS BUILT
00145
00146      MOVE WORKING-NAME          TO LISTING-NAME
00147      MOVE WORKING-REGULAR-HOURS TO LISTING-REGULAR-HOURS
00148      MOVE WORKING-OVERTIME-HOURS TO LISTING-OVERTIME-HOURS
00149
00150      *          CALCULATE TOTAL HOURS BY ADDING REGULAR, OVERTIME.
00151      *          NOTE THAT RESULT IS PLACED IN TIME-LISTING-LINE.
00152
00153      ADD WORKING-REGULAR-HOURS WORKING-OVERTIME-HOURS
00154      GIVING LISTING-TOTAL-HOURS
00155
00156      WRITE PRINT-LINE
00157      FROM TIME-LISTING-LINE
00158
00159      PERFORM 100-INPUT-A-RECORD
00160

```

AHEL FRED	40	03	043
BAKER SUE	30	00	030
CHARLIE CHUCK	40	12	052
FLECKSTEINER CINDY	40	00	040
FLECKSTEINER DON	40	20	060
FLECKSTEINER STACY	15	00	015
PERELMAN BARNEY	40	15	055
PERELMAN MIKE	03	00	003
NUMBER OF EMPLOYEES IS	008		

أسئلة مراجعة

- ١ - ٢ ماذا يعنى القول ان لغة الكويل هى لغة توثيق ذاتي ؟
- ٢ - ٢ ما الاجزاء الاربعة لبرنامج الكويل ؟
- ٣ - ٢ ما الغرض العام من جزء التعريف ؟
- ٤ - ٢ ما الغرض العام من جزء الاوساط ؟
- ٥ - ٢ ما الغرض العام من جزء البيانات ؟
- ٦ - ٢ ما مخزن العمل ؟
- ٧ - ٢ ما الجزء الذى يحدد التشغيل الفعلى المراد أدائه ؟
- ٨ - ٢ ما القسم فى برنامج الكويل ؟
- ٩ - ٢ ما مقاطع برنامج الكويل ؟
- ١٠ - ٢ لماذا يقترح أن تحتوى أسماء مقاطع جزء الاجراءات على ارقام متتالية كجزء من الاسم ؟
- ١١ - ٢ ما النقطة المرتبة ؟
- ١٢ - ٢ وضع المنطقة A والمنطقة B فى الكويل .
- ١٣ - ٢ ما العناصر التى يجب ان تبدأ كتابتها فى المنطقة A ؟
- ١٤ - ٢ وضع اهمية الترجيح داخل المنطقة B.
- ١٥ - ٢ وضع قواعد الاستمرارية فى الكويل .
- ١٦ - ٢ وضع قواعد كتابة برنامج الكويل .
- ١٧ - ٢ ما الاستخدامات الخاصة للعمود السابع فى الكويل ؟
- ١٨ - ٢ ما كلمات الكويل المحجوزة ؟
- ١٩ - ٢ ما قواعد كتابة الاسماء التى يعرفها المبرمج ؟
- ٢٠ - ٢ وضع وظيفة الاسطر الفارغة فى برنامج الكويل .

مسائل محلولة

- ٢ - ٢١ ما أنواع عناصر البيانات التى تنتمى الى مخزن العمل ؟
- يستخدم مخزن العمل لعناصر البيانات التى لا تكون حقولا فى سجلات ملفات . عادة ماتحسب مثل هذه العناصر بواسطة البرنامج ، او تستخدم كمناطق عمل مؤقتة اثناء التشغيل . وعادة ما يتحدد فى مخزن العمل مايلى :
- **عدادات** ، او مواقع ذاكرة تستخدم فى حفظ بيانات عددية تمثل عدد (مثل عدد العاملين) . عادة ماتوضع قيم ابتدائية مساوية للعدادات .
 - **مركبات** ، او مواقع ذاكرة تستخدم فى تركيب اجمالى (مثل اجمالى الموازنة المستحقة) . وعادة ماتوضع قيمة مساوية صفر للمركبات accumulators.
 - **اشارات و هغاتيخ للبرنأ هج** ، وهذه هى مواقع ذاكرة تحفظ معلومات، تحدد ما اذا كان احد الانشطة قد حدث اثناء تنفيذ البرنامج . عادة .. مايكون للاشارات flags والمفاتيخ switches قيم : صحيح true او خطأ false (مثل : اذا كانت كل سجلات الملف اجرى عليها التشغيل ام لا) . وتعد الاشارات وبعاد اعدادها كلما كانت هناك حاجة اثناء تنفيذ البرنامج .

• مناطق صور الأسطر . من المريح عادة عمل سطر معلومات في مخزن العمل، ثم نقل هذا السطر الى

مناطق سجل الملف طباعة قبل طباعته النهائية بعبارة write .

٢ - ٢٢ ماذا يفعل الكمبيوتر عندما ينفذ صيغة لغة الآلة من PERFORM 090- CALCULATE- TAX ؟

يتجه الكمبيوتر الى اول التعليمات في صيغة لغة الآلة المقطع المسمى 090- CALCULATE- TAX . ويستمر عند ذلك في تنفيذ كل التعليمات الاخرى الموجودة في المقطع ، واحدة تلو الاخرى ، واخيرا .. يعود الى التعليمة التي تلى PERFORM 090- CALCULATE- TAX مباشرة .

٢٣ - ٢٣ ماذا يفعل الكمبيوتر عندما ينفذ صيغة لغة الآلة من MOVE INVENTORY- ON- HAND- AMOUNT TO INVENTORY- REPORT- AMOUNT ؟

في كلمات عامة ينسخ الكمبيوتر محتويات موقع الذاكرة (الحقل الراسل) INVENTORY- ON- HAND- AMOUNT في موقع آخر للذاكرة (الحقل المستقبل) INVENTORY- REPORT- AMOUNT . وتظل قيمة INVENTORY- ON- HAND- AMOUNT دون تغيير . وتعتمد المحتويات الدقيقة للحقل المستقبل على وصف جزء البيانات العنصرين .

٢٤ - ٢٤ حدد ما اذا كانت الاسماء التالية غير صحيحة او صحيحة لكنها ضعيفة بالنسبة الى اسلوب البرمجة او انها مقبولة بالكامل .

INPUT (ا)

INPUT- INVENTORY- RECORD (ب)

B- INPUT- INVENTOY- RECORD- 040 (ج)

PROCESS- DATA (د)

(ا) غير صحيح : اسم يعرفه المستفيد ، يكرر كلمة محجوزة .

(ب) صحيح (يمكن تحسينها باضافة رقم متتالي للاسم) .

(ج) صحيح : خليط الحروف والارقام المتتالية مع اسم يصف مايفعله المقطع ، يكون اختياريا .

(د) صحيح : الا ان الاسم ضعيف ؛ حيث انه لا يصف مايفعله المقطع بالفعل .

٢٥ - ٢٥ وضح الاستخدامات الخاصة التالية للعمود السابع .

(a) MOVE SPACES TO WORKING-NAME-ADDRESS-LINE

(b) THE FOLLOWING PARAGRAPH COMPUTES STATE TAX

(c) MOVE "THE PENNSYLVANIA STATE UNIVERSITY BOOKSTORE" TO REPORT-HEADING ← col. 72

(d) D EXHIBIT NAMED NUMBER-OF-EMPLOYEES-PROCESSED
↑ ↑
col. 7 B Margin

(ا) حيث ان WORKING- NAME- ADDRESS- LINE مقسم على سطرين .. فستكون هناك حاجة الى وجود شرطة في العمود السابع؛ لتحديد الاستمرار . لاحظ امكانية تجنب الاستمرار على النحو التالي .

MOVE SPACES TO
WORKING-NAME-ADDRESS-LINE

(ب) النجمة الموجودة في العمود السابع تحدد إن بقية السطر عبارة عن تعليق ، يستخدم لتوضيح مايفعله جزء معين من البرنامج وكيفية عمله ذلك

(ج) تستخدم الشرطة الموجودة في العمود السابع لاستمرار سلسلة موضوعية بين علامتى تنصيص (ثابت غير عددي) من سطر للسطر التالي له . لاحظ انه لا توجد علامة تنصيص في نهاية السطر الاول (الذى ينتهى في العمود السابع) ، وانه توجد علامة تنصيص في بداية المنطقة B من السطر الثاني . وقد كان من الممكن تجنب هذا الاستمرار بكتابة مايلي :

MOVE

“THE PENNSYLVANIA STATE UNIVERSITY BOOKSTORE”
TO REPORT-HEADING

(د) تحدد D الموجودة في العمود السابع ان السطر هو سطر تصحيح ، وتستخدم في إيجاد اخطاء في البرنامج . (انظر رقم ٢ من الفصل الرابع) .

٢ - ٢٦ علق على الاسماء التى يعرفها المبرمج لعناصر البيانات التالية :

TNWI (i)

TOTAL- NET- WORTH- OF- INVENTORY- FOR- JULY (ب)

PHONE NUMBER (ج)

- ON- HAND (د)

ON- HAND (هـ)

(ا) يجب ان يكون اسم البيانات وصفيًا . وبالرغم من ان TNWI صحيح الا ان TOTAL- NET- WORTH- IN- VENTORY أفضل .

(ب) يجب ألا يزيد طول الاسماء التى يعرفها المبرمج عن ٣٠ رمزا ، وعلى هذا فالاسم خطأ .

(ج) يجب ألا توجد فراغات في الاسماء التى يحددها المبرمج (كما يجب ألا تشتمل على أى رمز آخر سوى الحروف الأبجدية من A الى Z ، والارقام من 0 الى 9 والشرطة فقط، وعلى هذا فالاسم خطأ .

(د) غير صحيح لوجود الرمز الخاص # في الاسم .

(هـ) لا يمكن ان ينتهى الاسم بشرطة «غير صحيح» .

مشاكل متكاملة

٢ - ٢٧ اعمل لقاءً مع مبرمج كويل له خبرة في البرمجة .

ناقش نمطيات كتابة الشفرة التى يستخدمها بالنسبة لـ (ا) اسطر التعليق، (ب) الاستمرارية . (ج) الاسماء التى يعرفها المبرمج . (د) النقطة المرتبة . (هـ) الترحيل . (و) الاسطر الفارغة .

٢ - ٢٨ أعد قراءة مثال ٢ - ١ وانتظر اذا كنت تستطيع ان تفهم معظم الموجود في البرنامج ام لا . هل تستطيع ان تفهم البرنامج بدون التعليقات الموجودة فيه ؟ او بدون الاسطر الفارغة ؟ او بدون الترحيل ؟ او بدون الاسماء الوصفية للبيانات؟

تمارين برمجة

فيما يلي ... كل المبالغ النقدية معطاة مقربة بالدولار (انظر الفصل الخامس لمعاملة الدولار والسنت) .
٢ - ٢٩ اكتب برنامجا كاملا بلغة الكوبل ، واختبره وصححه لطباعة التقرير التالي . اختر بيانات اختيارية، وتحقق من صحة المخرجات .

١ - طول سطر التقرير 132 خانة، ويشمل مايلي:

على التوالي

customer number - ٥ خانات ، حرفي عددي

١٠ خانات فارغة

customer name - ٢٠ خانة ، حرفي عددي

١٠ خانات فارغة

item numbe - ٧ خانات ، حرفي عددي

٥ خانات فارغة

quantity ordered - ٤ خانات ، عددي

٥ خانات فارغة

quantity shipped - ٤ خانات ، عددي

بقية السطر فارغ

٢ - بيانات المدخلات مثقبة في بطاقات على النحو التالي :

customer number - أول خمس اعمدة

customer name - الثلاثون عمود التالية

item number - السبع أعمدة التالية

quantity ordered - الأربع اعمدة التالية

quantity shipped - الأربع اعمدة التالية

بقية البطاقات - غير مستخدمة

٣ - أطلع في نهاية التقرير سطرا يبين اجمالي عدد العناصر المطلوبة items ordered (لكل العملاء) . واجمالي عدد

العناصر المرسله items shipped (لكل العملاء)

٢ - ٣٠ اكتب برنامجا كاملا بلغة الكوبل واختبره وصححه لطباعة التقرير التالي :

customer number	فراغ	old balance	فراغ	payment amount	فراغ	new balance
٨	٥	٥	٥	٤	٥	٥
خانات	خانات	خانات	خانات	خانات	خانات	خانات
حرفي عددي		عددي		عددي		عددي

والمدخلات مثقبة على بطاقات على النحو التالي :

customer number	old balance	payment amount	بقية الخانات (٦٣ خانة) غير مستخدمة
٨ خانات	٥ خانات	٤ خانات	

تحتسب الموازنة الجديدة new balance بطرح ماتم دفعه payment من الموازنة القديمة old balance .

٢ - ٢١ أكتب برنامجاً كاملاً بلغة الكوبل، واختبره وصححه لطباعة التقرير التالي :

تحتوى المخرجات على الحقول التالية مع ترك مسافات مناسبة :

part number - ٨ خانات ، حرفى عددي

on hand - ٥ خانات ، عددي

on order - ٥ خانات ، عددي

quantity - ٦ خانات ، عددي

المدخلات مثقبة فى بطاقات على النحو التالي :

part number - أول ٨ أعمدة

on hand - الخمس أعمدة التالية

on order - الخمس أعمدة التالية

بقية البطاقات فارغة

تحتسب الكمية المتاحة quantity available بجمع الكمية الموجودة on hand على الكمية المطلوبة on order .

أطبع فى النهاية سطوراً يبين إجمالى الكمية المطلوبة (لكل الأجزاء) .

٢ - ٢٢ أكتب برنامجاً كاملاً بلغة الكوبل واختبره وصححه لطباعة التقرير التالي :

المخرجات (مع ترك المسافات المناسبة بينهما) .

general ledger account number - ٤ خانات ، حرفى عددي

description - ٢٠ خانة ، حرفى عددي

credit amount - ٥ خانات ، عددي

debit amount - ٥ خانات ، عددي

المدخلات مثقبة فى بطاقات ومرتبة ؛ طبقاً لرقم حساب دفتر الأستاذ العام general ledger account number ،

وموجودة على النحو التالي :

general ledger account number - ٤ خانات

description - الخانات العشرين التالية

credit amount - الخانات الخمس التالية

debit amount - الخانات الخمس التالية

بقية البطاقة - غير مستخدمة

أطبع في النهاية سطرا ، يبين إجمالي الرصيد الدائن credit وإجمالي الرصيد المدين debit .

٢ - ٢٢ أكتب برنامجا بلفة الكوبل، واختبره وصححه لطباعة القائمة التالية من ملف العاملين الرئيسى :

تحتوى المخرجات على الحقول التالية مع وجود خمس خانات فارغة بين كل حقل وآخر :

employee ID - ٦ خانات ، حرفى عددى

number of deductions - خانتان ، عددى

last year's annual salary - ٥ خانات ، عددى

this year's annual salary المدخلات مثقبة فى بطاقات على النحو التالى :

employee ID - الاعمدة من رقم ١ الى رقم ٦

annual salary - ٥ خانات ، عددى

number of deductions - العمودان رقم ٧ - ٨

annual salary last year's - الاعمدة من ٩ - ١٣

this year's annual salary - الاعمدة من رقم ١٤ - ١٨

أطبع في النهاية سطراً يبين إجمالى الرواتب لهذا العام ، للعام الماضى ، وعدد العاملين الذى أجرى لهم تشغيل .

٢ - ٢٤ أكتب برنامجا بلفة الكوبل، واختبره وصححه لطباعة القائمة التالية من ملف حسابات الدائنين :

المخرجات (مع ترك ٨ خانات فارغة بين كل حقل وآخر) :

invoice number - ٤ خانات ، حرفى عددى

invoice date (mm yy dd) - ٦ خانات ، عددى

invoice amount - ٥ خانات ، عددى

discount amount - ٤ خانات ، عددى

amount due (less discount) - ٥ خانات ، عددى

المدخلات مثقبة فى بطاقات على النحو التالى

invoice number - الاعمدة من ١ - ٤

invoice amount - الاعمدة ٥ - ٩

discount amount - الاعمدة ١٠ - ١٣

invoice date (mm yy dd) - الاعمدة ١٤ - ١٨

أطبع في النهاية سطرا ، يبين إجمالى قيمة الفاتورة، وإجمالى الخصومات وإجمالى القيمة مطروحا منه الخصومات .

٢ - ٢٥ أطبع تقريراً يبين الحقول التالية من ملف طلبية احد الجامعات :

Student # Name Year # Credits Completed

المدخلات مثقبة فى أعمدة بطاقات على النحو التالى :

student number - الأعمدة ١ - ٩

name - الأعمدة ١٠ - ٢٩

year (4, 3, 2, 1) - العمود رقم ٣٠

أطبع فى النهاية سطرا بإجمالى عدد الساعات التى أنهيت وإجمالى عدد الطلبة .

٣٦ - ٢ أطبع عناوين بريدية لها الشكل التالى :

M. MOUSE
1031 EDGECOMB AVE.
YORK, PA 17403

المدخلات مثقبة فى أعمدة بطاقات على النحو التالى :

name - الأعمدة ١ - ٢٠

street - الأعمدة ٢١ - ٤٠

city - الأعمدة ٤١ - ٥١

state - الأعمدة ٥٢ - ٥٣

zip code - الأعمدة ٥٤ - ٥٨

ملاحظة : (١) أنت فى حاجة الى ثلاث مناطق تخيلية لسطر المخرجات (المشكلة رقم ٢١ من هذا الفصل) .

(٢) كل سجل مدخلات ينتج عنه ثلاثة أسطر مخرجات.

(٣) استخدم .

WRITE PRINT-LINE
FROM WORKING-NAME-LINE
AFTER ADVANCING 2 LINES

فى ترك سطر فارغ بين العناوين

٣٧ - ٢ أكتب برنامجاً لطباعة شيكات ، لها الشكل التالى :

_____ PAYEE	_____ CHECK #
	_____ AMOUNT (dollars)

المدخلات متقبة فى بطاقات وتحتوى على اسم المستفيد من الشيك payee فى الأعمدة (١ - ٢)، وقيمة الشيك فى الأعمدة (٢١ - ٢٥). ينتج البرنامج أرقام الشيكات بدءاً برقم الشيك 100 # ورقم الشيك الذى يليه هو 110، ثم 120.. وهكذا. بعد طباعة كل الشيكات، أطيح سطراً يبين إجمالى المبالغ المدفوعة disbursed.

٢ - ٣٨ أكتب برنامجاً يطبع تقريراً بنكيًا من مدخلات البطاقة التالية :

customer name - الأعمدة (١ - ٢٠)

saving account balance - الأعمدة (٢١ - ٢٥)

checking account balance - الأعمدة (٢٦ - ٣٠).

certificates of deposit balance - الأعمدة (٣١ - ٣٥)

يجب أن تحتوى المخرجات على سطرين لكل عميل يبين اسم العميل والموازنات المختلفة له. وفى النهاية.. يطبع سطر يبين اجمالى موازنات المدخرات total saving و اجمالى الشيكات total checkinzg و اجمالى موازنات الشهادات total certificate sbalances. التخطيط الدقيق للتقرير متروك لك.

الفصل الثالث

جزء التعريف

Identification Division

٣ - ١ ترهيز التكوين

لايسرى هذا القسم على جزء التعريف فقط ، بل يسرى كذلك على بقية برنامج الكويل أيضا .
تشير كلمة تكوين syntax إلى التكوين النحوى للغة . مثال ذلك ، الجملة الانجليزية "dog the chaxed cat the" بها
عديد من الاخطاء التكوينية syntax errors تشمل الهجاء وترتيب الكلمات . إذا أحتوى برنامج كويل اخطاء تكوينية ، فلن
يستطيع المترجم ان يترجم الجملة (الجمل) الفردية ، وان يتواجد برنامج تنفيذ لتنفيذه . وتساعد رسائل الخطأ التشخيصية ،
والتي تطبع كجزء من قائمة برنامج المصدر ، المبرمج فى ايجاد الاخطاء التكوينية وتصحيحها .

وقد طور ترميز شائع الاستخدام لتوضيح التكوين الصحيح لجمل الكويل ، وتستخدم فيه الاصطلاحات التالية .

(١) الأقواس المربعة [] تحتوى على مجموعة من عنصر واحد او أكثر اختيارية optional . تستطيع أن تختار أى واحد من
المجموعة أو لا تأخذ شيئا على الإطلاق .

(٢) الأقواس { } تحتوى على مجموعة من عنصر واحد او اكثر . ويجب ان تختار واحدا بالضبط exactly one من هذه
العناصر .

(٣) حروف كبيرة Uppercase leters تستخدم للكلمات المحجوزة ، وتستخدم الحروف الصغيرة lowercase letter
لل كلمات التى يعرفها المبرمج .

(٤) يجب أن تظهر الكلمات المحجوزة التى يوضع تحتها خط على حالها فى العنصر ، اما الكلمات المحجوزة الأخرى .. فيمكن
حذفها .

(٥) النقاط (...) تستخدم لتعريف عناصر يمكن ان تتكرر اى عدد من المرات .

مثال ٣ - ١

اعتبر الترميز التالى لعبارة MOVE من جزء الإجراءات .

MOVE: { identifier-1
literal } TO identifier-2 [identifier-3] ...

يوضع خط تحت كلمة MOVE، ولا تظهر الكلمة بين قوسين مربعين ، وعلى هذا .. فلا بد من ظهورها . ويستطيع المبرمج أن يستخدم أما معرفاً Identifier أو ثابتاً literal (الثوابت التي تعرضنا لها حتى الآن هي السلاسل الموجودة بين علامتي تنصيص فقط) ؛ حيث إنها بالحروف الصغيرة فيقوم المبرمج بتحديد المعرف (أو الثابت) المطلوب ، والذي يتبعه المعرف المطلوب (الذي يحدده المبرمج) . ويمكن أن يتبع هذا بدوره عددا (يشمل لاشيء) من المعرفات التي يعرفها المبرمج .

مثال ٣ - ٢ :

نعلق على تحقيقات عديدة لعبارة مثال ٢ - ١

- MOVE INPUT-CUSTOMER-NAME TO OUTPUT-CUSTOMER-NAME

صحيحة : فهي في صيغة 2 - identifier To identifier-1

- MOVE "OVERPAID" TO INVOICE-MESSAGE-AREA TYPE-OF-ACCOUNT

صحيحة : فهي في صيغة 3 - identifier To literal

- MOVE DISCOUNT-AMOUNT TO INVOICE-DISCOUNT AND SHIPPING-DISCOUNT

غير صحيحة : يجب ألا تظهر AND بين المعرفات ، INVOICE- DISCOUNT, SHIPPING- DISCOUNT

- MOVE GREETING TO "HELLO"

غير صحيحة : بالرغم من أن "HELLO" ثابت صحيح .. ألا أنه من المنوع ظهور ثابت بعد كلمة TO .

- MOVE "HELLO" GROSS-SALES TO GREETING SALES-TOTAL

غير صحيحة : يظهر معرف واحد فقط ، أو ثابت واحد فقط بين MOVE و TO .

٣ - ٢ تكوين جزء التعريف

جزء التعريف الذي سبق ذكر عمله في القسم الأول من الفصل الثاني - الشكل التكويني المبين في شكل ٣ - ١

IDENTIFICATION DIVISION.
PROGRAM-ID. program-name.
[AUTHOR. comment.]
[INSTALLATION. comment.]
[DATE-WRITTEN. comment.]
[DATE-COMPILED.]
[SECURITY. comment.]

شكل (٣ - ١)

لاحظ أن عنوان الجزء (اسم الجزء) ، ومقطع PROGRAM- ID هما الذى يجب ظهورهما فى هذا الجزء، أما بقية المقاطع فظهورها اختياري .

مثال ٣ - ٢ :

قارن الاسطر من رقم ١ الى رقم ٨ فى مثال ١٠ ٢ بالنموذج التكويني الموجود فى شكل ١٠ ٢ .

٣ - ٣ مقطع تعريف البرنامج

يستخدم مقطع تعريف البرنامج PROGRAM- ID لتحديد اسم لبرنامج الهدف . وعندما يوضع هذا البرنامج فى صورته النهائية فى ملف قرص (فى مكتبة البرامج) ... يعرف باسمه الموجود فى هذا المقطع (والذى يحفظ فى دليل المكتبة) ، وتحت هذا الاسم يمكن الاتصال بالبرنامج لتنفيذه بون تعقيدات .

يجب أن يتفق هذا الاسم بصفة عامة مع قواعد كتابة الاسماء التى يعرفها المبرمج (القسم الخامس من الفصل الثانى) .
 إلا أن مترجم الكويل - فى نظم IBM OS/VS وتنظم أخرى ، يحول تعريف البرنامج بالكويل الى اسم من نوع اسماء برامج المكتبة ويحدث ذلك تلقائيا . وبالنسبة الى كويل IBM OS/VS : (١) يجب ألا يزيد الاسم عن ٨ خانات ، (٢) يجب ألا يحتوى الا على حروف هجائية وارقام فقط ، (٣) يجب ان يكون اول رمز حرفا هجائيا .

مثال ٢ - ٤

• INVENTORY-REPORT سوف تصبح INVENTOR

(مسموح بثمان خانات فقط ، يحذف المترجم بقية الحروف) .

• INVRPT لا يحدث لها أى تغيير .

• PRINT-PAYCHECKS سوف تصبح PRINT 0 PA

(مسموح بثمان خانات فقط ، غير مسموح بالشرطة ، وعلى هذا يحولها المترجم إلى 0) .

• 1- UPDATE- RECIEVABLE سوف تصبح A 0 UPDATE

(مسموح بثمان خانات فقط ، غير مسموح بالشرطة ، وعلى هذا يحولها المترجم إلى O) ، غير مسموح بالبداية برقم وعلى هذا يستبدله المترجم بالحرف A) .

• RECVUPDT لا يحدث أى تغيير .

٣ - ٤ مقاطع اختيارية

جميع المقاطع المتبقية فى جزء التعريف اختيارية ، وتعامل الجمل الموجودة فى كل مقطع منها على أنها تعليقات . ونظراً لأن هذه المقاطع قد تحتوى على أى شئ يريد المبرمج كتابته ، فيجب أن تعكس الاسماء نوع البيانات المقدمة بوضوح .

مثال ٢ - ٥ :

- INSTALLATION. RESEARCH AND DEVELOPMENT.

يحدد مقطع INSTALLATION مكان مركز الكمبيوتر الذى أعد فيه البرنامج (يمكن أن يكون بالمنشأة الكبيرة عديد من نظم الكمبيوتر ، المنتشرة فى مواقع جغرافية مختلفة) .

- DATE-COMPILED.

مقطع DATE-COMPILED خاص لأن المترجم ، وليس المبرمج ، يتولى ماؤه .

- SECURITY. AUTHORIZED PERSONNEL ONLY—SEE POLICY MANUAL 301.

يحدد مقطع SECURITY أى اعتبارات أمن تسرى على هذا البرنامج أو على الملفات التى يقوم بتشغيلها .

باتباع المقاطع المختلفة لجزء التعريف.. يوصى - بشدة - أن يدخل المبرمج مجموعة من بطاقات الاوامر (مع وجود نجمة فى العمود السابع) معطيا وصفا موجزا لما يفعله البرنامج وهذا يوفر الوقت والمال عندما يأتى مبرمج آخر فيما بعد ليدخل تغييرات على هذا البرنامج (وهذا يسمى ببرمجة صيانة maintenance programming) . يهدف التوثيق الجيد للبرنامج إلى جعل البرنامج أسهل فى قراءته وفهمه وتعديله .

مثال ٣ - ٦ :

انقد جزء التعريف التالى لبرنامج ينفذ على نظام كمبيوتر IBM OS/VS :

PROGRAM-ID. PRINT-MAILING-LABELS.
DATE-WRITTEN. OCT 1983.

تقنيا ... الشئ الوحيد الخطأ هو أن عنوان الجزء (IDENTIFICATION DIVISION) غير موجود إلا أنه :

(١) PRINT MAILING - LABELS ليست فى صورة مكتبة برامج ، ويحولها المترجم إلى PRINT MAILING - LABELS .

(٢) حذف DATE-COMPILED, INSTALLATION, AUTOUR و SECURITY ، وعبارات التعليق معطيا عرضا موجزا لما يفعله البرنامج، بما يمثل برمجة ضعيفة . التوثيق الجيد هو جزء أساسى لى برنامج كوبر .

أسئلة مراجعة

- ١-٣ ما الغرض من جزء التعريف ؟
- ٢-٣ عرف (أ) تكويني (ب) خطأ تكويني (ج) رسائل تشخيصية ؟
- ٣-٣ وضع ترميز التكوين المستخدم في هذا الكتاب .
- ٤-٣ اذكر تكوين جزء التعريف .
- ٥-٣ ما الأجزاء التي يجب ظهورها في جزء التعريف ؟ وما الأجزاء الاختيارية ؟
- ٦-٣ وضع قواعد كتابة جزء التعريف ؟
- ٧-٣ ما وظيفة مكتبة البرامج ؟
- ٨-٣ ما قواعد مقطع تعريف البرنامج في نظم IBM ؟
- ٩-٣ وضع ما المعلومات التي تعطى في كل من المقاطع الاختيارية من جزء التعريف .
- ١٠-٣ ماهي برمجة الصيانة ؟ كيف ترتبط بجزء التعريف ؟

مسائل محلولة

- ١١-٣ بمعرفة النموذج التالي لعبارة PERFORM من جزء الإجراءات

PERFORM { paragraph-name-1 } [{ THROUGH } { paragraph-name-2 }]
 { section-name-1 } [{ THRU } { section-name-2 }]
 { identifier-1 } **TIMES**
 { integer-1 }

حدد ما إذا كان مايلي صحيحا تكوينيا، أم لا :

- (a) PERFORM CALCULATE-DISCOUNT-AMOUNT 10 TIMES
- (b) PERFORM READ-EMPLOYEE-RECORD THROUGH PRINT-PAYCHECKS
NUMBER-OF-EMPLOYEES TIMES
- (c) PERFORM CHECK-DEPENDENT-STATUS TO COMPUTE-DEDUCTIONS
NUMBER-OF-DEPENDENTS TIMES
- (d) PERFORM THRU VALIDATE-ZIP-CODE 9 TIMES
- (e) PERFORM CHECK-DEPARTMENT-CODE THROUGH CODE-COUNT TIMES
- (f) PERFORM PRINT-INVOICE-LINE 7

(أ) صحيح

(ب) صحيح

(ج) غير صحيح : يجب استخدام THROUGH أو THRU وليس TO

(د) غير صحيح : أي من : section- name-1 أو paragraph- name-1 مطلوب :

. PERFORM VALIDATE- NAME THRU VALIDATE- ZIP- CODE 9 TIMES

(هـ) غير صحيح : أي من paragraph- name-2 أو section- name-2

يجب أن يتبع THRU أو THROUGH :

PERFORM CHECK- DEPARTMENT- CODE THRU PRINT- DEPARTMENT-
.TOTAL CODE- COUNT TIME.S

(و) غير صحيح : TIMES مطلوبة :

. PERFORM PRINT- INVOICE- LINE 7 TIMES

٣ - ١٢ أى مما يلي يعتبر تعريف برنامج مناسب :

(a) MONTHLY-SALES-REPORT

(b) PRINT-ACCOUNTS-RECEIVABLE-AGED-TRIAL-BALANCE

(أ) صحيح : الاسم وصفى بطريقة مناسبة لما يفعله البرنامج . يحول مترجم كويل IBM الاسم الى 0 MONTHLY ، وعلى هذا يكون من الأفضل استخدام اختصار من ٨ خانات فى المواقع الأولى مثل SALESRPT .

(ب) غير صحيح : يجب أن يكون تعريف البرنامج اسما يعرفه المبرمج (محدد بعدد ٢٠ خانة) .

٣ - ١٢ أكتب جزء التعريف من برنامج كويل لطباعة شيكات . كل المعلومات الخاصة بالشيكات موجودة فى ملف قرص ، وسبق إنتاجها ببرنامج رواتب آخر .

فى الحل التالى .. لاحظ استخدام أسطر فارغة لتوضيح تعريف البرنامج .

IDENTIFICATION DIVISION.

PROGRAM-ID. PRNTCHCK.

AUTHOR. MIKE PERELMAN, JUNIOR PROGRAMMER, PAYROLL APPLICATIONS.

INSTALLATION. XYZ COMPANY, BALTIMORE, MD.

DATE-WRITTEN. 7 SEPT., 1983.

DATE-COMPILED.

SECURITY. SEE POLICY MANUAL 7A3 REGARDING HANDLING OF
COMPANY CHECKS.

- * PRNTCHCK PRINTS EMPLOYEE PAYCHECKS BY COPYING THE PAYROLL
- * INFORMATION ALREADY ON DISK FILE PAYCHECK.IMAGE ONTO
- * PRE-PRINTED CHECKS IN THE PRINTER. THE OPERATOR HAS
- * THE CAPABILITY TO STOP OR BACK-UP PRINTING IN CASE OF
- * DIFFICULTY WITH THE FORMS AND/OR PRINTER. PRNTCHCK ALSO
- * LOGS THE FOLLOWING CONTROL INFORMATION: # CHECKS PRINTED,
- * # CHECKS BACKED-UP, TOTAL \$ AMOUNT OF CHECKS.

تعاريف بومجة

من ٣ - ١٤ إلى ٣ - ٢٢ عدل التعاريف من رقم ٢ - ٢٩ / ٢ - ٢٨ فى الفصل الثانى، لتعكس ما تعلمته الآن عن جزء التعريف .

الفصل الرابع

جزء الأوساط

Environment Division

٤ - ١ تكوين جزء الأوساط

لجزء الأوساط الذى ذكرت وظيفته العامة فى القسم الاول من الفصل الثانى، الشكل التكويني المبين فى شكل (٤ - ١) (أدخلت ارقام الأسطر لتسهيل الإشارة إليها) . ويمكن رؤية مكونات جزء الأوساط، وهى : قسم تشكيل CONFIGURATION SECTION ، وقسم مدخلات ومخرجات INPUT- OUTPUT SECTION . ويشمل القسم الأول مقاطع كمبيوتر المصدر SOURCE- COMPUTER وكمبيوتر الهدف OBJECT- COMPUTER ، وأسماء خاصة SPECIAL- NAMES . أما القسم الثانى.. فيشمل مقطعا واحدا فقط ، وهو التحكم فى الملفات FILE- CONTROL . تذكر ان عنوان الجزء (السطر الأول من شكل ٤ . ١) ، وعناوين الأقسام (السطران رقم ٢ ، ١٤) ، وعناوين المقاطع (الأسطر : ٣ و ٤ و ٦ و ١٥) ، يجب أن تبدأ كتابة بقية محتوياتها فى المنطقة B . أحيانا تكتب محتويات المقطع على نفس السطر المكتوب فيه عنوان المقطع، وأحيانا لاتكتب. يجب أن يكون المظهر المرئى الجيد لديك فى البرمجة .. (انظر مثال ٤ - ١) . ولاتناقش هنا الا المحتويات التى تسرى على النظام التابعى للملفات .

مثال ٤ - ١ :

```
00031 ENVIRONMENT DIVISION.
00032
00033 CONFIGURATION SECTION.
00034
00035 SOURCE-COMPUTER. IBM-370 WITH DEBUGGING MODE.
00036 OBJECT-COMPUTER. IBM-370
00037 PROGRAM COLLATING SEQUENCE IS
00038 CUSTOMER-NUMBER-CODE-SEQUENCE.
00039 SPECIAL-NAMES.
00040
00041 COL IS TOP-OF-PAGE
00042 CUSTOMER-NUMBER-CODE-SEQUENCE IS
00043 "3" THRU "9"
00044 "2"
```

```

00045          "0" THRU "1"
00046
00047
00048          INPUT-OUTPUT SECTION.
00049
00050          FILE-CONTROL.
00051
00052              SELECT CUSTOMER-SALES-FILE
00053                  ASSIGN TO CUSTSALE
00054                  ORGANIZATION IS SEQUENTIAL
00055                  ACCESS IS SEQUENTIAL
00056                  FILE STATUS IS CUST-SALES-FILE-STATUS
00057
00058
00059              SELECT SALES-REPORT
00060                  ASSIGN TO SALERPT
00061                  ORGANIZATION IS SEQUENTIAL
00062                  ACCESS IS SEQUENTIAL
00063                  FILE STATUS IS SALES-REPORT-STATUS
00064

```

- (1) ENVIRONMENT DIVISION.
- (2) CONFIGURATION SECTION.
- (3) SOURCE-COMPUTER. computer-name [WITH DEBUGGING MODE].
- (4) OBJECT-COMPUTER. computer-name.
- (5) [PROGRAM COLLATING SEQUENCE IS alphabet-name].
- (6) SPECIAL-NAMES.
- (7) [function-name IS cobol-name] ...
- (8) [alphabet-name IS
- (9) [STANDARD-1
- (10) [NATIVE
- (11) [literal-1 [{ THROUGH } literal-2] ...
- (12) [[THRU]
- (13) [[ALSO literal-3 [ALSO literal-4]] ...
- (14) [[THROUGH } literal-6]] ...
- (15) [[THRU]
- (16) [[ALSO literal-7 [ALSO literal-8]]]]]
- (17) .]
- (18) INPUT-OUTPUT SECTION.
- (19) FILE-CONTROL.
- (20) SELECT file-name
- (21) ASSIGN TO external-name
- (22) [ORGANIZATION IS SEQUENTIAL]
- (23) [ACCESS MODE IS SEQUENTIAL]
- (24) [FILE STATUS IS data-name]
- (25) [RESERVE integer [AREAS]
- (26) [AREA]]

شكل (٤ - ١)

٢ - ٢ قسم التشكيل :

مقطع كمبيوتر المصدر

يصف قسم التشكيل نظام الكمبيوتر المستخدم في ترجمة البرنامج وتنفيذه . ويبدأ مقطع كمبيوتر المصدر -SOURCE COMPUTER باسم الكمبيوتر الذي يحدد الكمبيوتر المستخدم في الترجمة . وتعامل محتويات المقطع كتعليق ولا تخدم إلا كتوثيق فقط للبرنامج . وتختلف صيغة اسم الكمبيوتر من نظام كمبيوتر لنظام آخر : وبالنسبة لنظم كمبيوتر IBM طراز 370 تكفى كتابة IBM- 370 .

هناك محتوى اختياري كذلك لمقطع كمبيوتر المصدر وهو WITH DEBUGGING MODE (السطر الثالث في شكل ٤ - ١) . فإذا كتب هذا الجزء تترجم كل أسطر التصحيح (والمحدد لها الحرف D في العمود السابع منها) إلى برنامج الهدف وينفذها على ذلك الكمبيوتر أثناء تنفيذه برنامج الهدف . فإذا لم يوجد هذا الجزء . تعامل أسطر التصحيح مثل التعليقات (أى إنه بالرغم من طباعتها في قائمة برنامج المصدر ، إلا أنها لا تترجم إلى برنامج الهدف وعلى هذا لا يكون لها أى تأثير على التنفيذ) . ويعالج التصحيح بالكامل في الفصل التاسع .

مثال ٤ - ٢ :

- SOURCE-COMPUTER. IBM-370-158.

يضاف طراز الكمبيوتر رقم 158 اختياريًا إلى IBM 370 وحيث إنه لم يتحدد WITH DEBUGGING MODE ، تعامل العبارات الموجودة حرف D في عمودها السابع كتعليقات ، وعلى هذا لا يكون لها أى تأثير على التنفيذ . ويعامل اسم الكمبيوتر كتعليق .

- SOURCE-COMPUTER. IBM-370
WITH DEBUGGING MODE.

تترجم أسطر التصحيح إلى برنامج الهدف (مثل العبارات المعتادة) وتنفذ . ويكتب جزء WITH DEBUGGING MODE في سطر خاص به لتسهيل رؤيته .

٢ - ٣ قسم التشكيل :

مقطع كمبيوتر الهدف

يحدد مقطع كمبيوتر الهدف OBJECT- COMPUTER الكمبيوتر المستخدم في تنفيذ برنامج الهدف . ويخدم اسم الكمبيوتر كتوثيق للبرنامج أساسا ، وعادة مايكون هو نفس اسم الكمبيوتر المستخدم في مقطع كمبيوتر المصدر .

تذكر ان الكمبيوتر يخزن معلومات عن طريق تمثيل كل رمز حرفي عددي برقم ثنائي فريد . عندما تتراوح هذه الأرقام من أقلها إلى أكبرها ... تنتج ترتيبا للرموز المناظرة ، يسمى تسلسل التتابع collating sequence . ويعتمد تسلسل التتابع ، بالطبع ، على الشفرة الثنائية المستخدمة ، ويوضح تسلسل التتابع لكل من شفرة EBCDIC ، وشفرة ASCII في ملحق ب .

مثال ٤ - ٣ :

في تسلسل التتابع لشفرة EBCDIC :

space < \$ < # < A < B < Z < 0 < 9

نفس الرمز مرتبة في تسلسل التتابع لشفرة ASCII على النحو التالي :

space < # < \$ < 0 < 9 < A < B < Z

تحتفظ كل من شفرة EBCDIC ، وشفرة ASCII بنفس الترتيب الأبجدي المعتاد أي أن "ZAG" < "ZIG".

ويوجد جزء اختياري كذلك في مقطع كمبيوتر الهدف ، (السطر الخامس من شكل ٤ - ١) والذي يسمح للمبرمج باختيار تسلسل تتابع لاستخدامه أثناء تنفيذ البرنامج . فإذا حذف هذا الجزء من البرنامج .. يسرى التسلسل التقليدي لنظام الكمبيوتر (وحيث إن التسلسل التقليدي غالبا ما يكون هو تسلسل التتابع المرغوب فيه .. فعادة ما يحذف هذا الجزء) . يحدد تسلسل التتابع النتائج عندما تقارن عناصر حرفية عديدة ، أو عندما تخزن هذه العناصر .

مثال ٤ - ٤ :

ارجع الى مثال ٤ - ١ . في مقطع كمبيوتر الهدف (الاسطر من ٣٦ - ٣٨) يعامل IBM-370 كتعليق ، بينما يدعى CUSTOMER- NUMBER- CODE- SEQUENCE بتسلسل تتابع يستخدم في البرنامج . وهذا التسلسل معرف بواسطة الاسطر من (٤٢ - ٤٥) من مقطع الأسماء الخاصة ليكون كما يلي :

3 < 4 < 5 < 6 < 7 < 8 < 9 < 2 < 0 < 1

٤ - ٢ قسم التشكيل :

مقطع الأسماء الخاصة

لهذا المقطع ثلاث وظائف رئيسية يمكن أن تعالج مستقلة عن بعضها البعض .

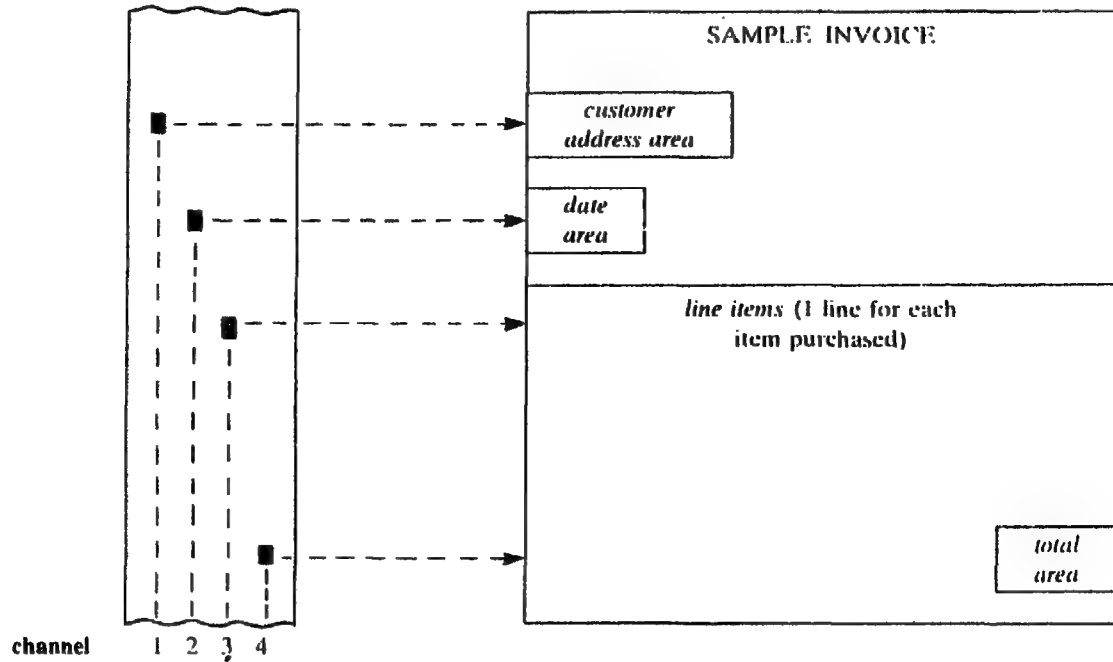
تسمية قنوات تحكم العربية

عندما تطبع المخرجات على اوراق مستمرة، او صيغ مستمرة continued forms ... تعرف مواقع عناصر البيانات المختلفة في الصيغة عن طريق قنوات تحكم العربية carriage control channels. وهذه القنوات محددة في شكل ٤ - ٢ كتقريب في شريط الورق الذي يتحرك مع الصيغة الأكثر احتمالا، في الطابعات الحديثة ... يمكن تحقيق أوضاع القناة إلكترونيا . وفي أي حالة من الحالات ، يستخدم السطر السابع من شكل ٤ - ١ في تحديد اسم كويل لقناة تحكم العربية للطابع . وفي كويل IBM OS/VS يحتوى الاسم الوظيفي لقناة على الحرف C ، يليه رقميان كرقم للقناة : C01 ، C02 و ... و C12 . يمكن أن يكون أي اسم يضعه المبرمج اسم كويل للقناة .

مثال ٤ - ٥ :

C01 IS TOP-OF-INVOICE
C02 IS INVOICE-DATE-AREA
C03 IS CUSTOMER-ADDRESS-AREA
C04 IS LINE-ITEM-AREA.

على هذا .. فالقنوات من ١ الى ٤ تصاحب الاسماء التي يعرفها المبرمج، والتي تستخدم فيما بعد في جزء الاجراءات للتحكم في موقع المخرجات المطبوعة . لاحظ إنه لا توجد الا نقطة واحدة فقط لكل مقطع الاسماء الخاصة (السطر ١٢ من شكل ٤ - ١) - لا تستخدم نقاط بين المحتويات . لاحظ كذلك الرمز الوصفي للاسماء التي يضعها المبرمج .



شكل (٤ - ٢)

تعريف تسلسل التتابع

إذا ما تحدد تسلسل تتابع غير تقليدي في مقطع كمبيوتر الهدف ، فيجب أن يعرف تسلسل التتابع هذا في مقطع الاسماء الخاصة (الاسطر من رقم ٨ الى رقم ١٢ من شكل ٤ - ١) . يجب أن يكون الاسم الأبجدي (السطر الثامن) في مقطع الاسماء الخاصة مثل الاسم الموجود في مقطع كمبيوتر الهدف (السطر الخامس) . وعلى هذا .. يعرف تسلسل التتابع المصاحب للاسم الأبجدي كأحد الاشياء التالية :

STANDARD-1 •

(السطر رقم ٩ من شكل ٤ - ١) يحدد تسلسل التتابع لشفرة ASCII

• NATIVE

(السطر رقم ١٠ من شكل ٤ - ١) يحدد تسلسل التتابع لشفرة EBCDIC ، وهي التقليدية لنظم كمبيوتر IBM من طراز (370) .

• اسم يعرفه المبرمج

(السطران رقم ١١ ، ١٢ من شكل ٤ - ١) يستطيع المبرمج أن يعرف تسلسل تتابع خاصاً به عن طريق كتابة سلسلة من الرموز بالإيجاز (literal-1 THROUGH literal-2) ، وتحدد تساوى رمزين بالبديل ALSO .

مرة أخرى نشير إلى إنه بغض النظر عن عدد قنوات تحكم العربية، وتسلسلات التتابع المعرفة فى مقطع الأسماء الخاصة .. فليست هناك سوى نقطة واحدة فقط ، فى نهاية المقطع كله .

مثال ٤ - ٦ :

OBJECT-COMPUTER. IBM-370 COLLATING SEQUENCE IS RUSH-ORDER-CODES.
SPECIAL-NAMES.

C01 IS TOP-OF-FORM
C02 IS MAILING-ADDRESS-AREA

RUSH-ORDER-CODES IS

"I" THRU "N"
"D" "G" "F" "E"
"A" THRU "C"
"H" ALSO "O"

هنا تعرف الاسماء الخاصة تسلسل التتابع المكتسب RUSH- ORDER- CODE كما يلى :

I < J < K < L < M < N < D < G < F < E < A < B < C < H = O

تعتبر H و O متساويتان فى التتابع. يفترض فى أى رموز غير محددة انها تأتى فى النهاية (بعد H و O) وتكون فى نفس ترتيبها الطبيعي، أى أن P تتبع H و O.

تسمية SYSOUT, SYSIN ، والشاشة المرئية للمشغل الآدهى

يمكن أن يحدد مقطع الاسماء الخاصة اختياريًا (السطر السابع من شكل ٤ - ١) أسماء كويل للغات نظام متخصص معين ، يتم تشغيلها بعبارات DISPLAY, ACCEPT من جزء الإجراءات (أنظر الفصل السادس) . تختلف التفاصيل الدقيقة لهذه الآلية من جهاز كمبيوتر، ونظام تشغيل معين لجهاز ونظام تشغيل آخر ، إلا أنه فى كويل IBM OS/VS .. يدخل أمر ACCEPT سجلاً واحداً من النهاية الطرفية لمشغل الكمبيوتر operator's console او من الملف SYSIN . ويخرج امر DISPLAY سجلاً واحداً إلى النهاية الطرفية لمشغل الكمبيوتر، أو الى الملف SYSOUT .

مثال ٤ - ٧ :

SPECIAL-NAMES.
 C01 IS TOP-OF-INVOICE
 CONSOLE IS OPERATOR-MESSAGE-DEVICE.

الاسم الوظيفي CONSOLE هو كلمة من كلمات IBM المحجوزة ، OPERATOR- MESSAGE- DEVICE هو الاسم الذي تشير بقية برنامج مصدر كويل إلى النهاية الطرفية الخاصة بمشغل الكمبيوتر في عبارات DISPLAY :ACCEPT

DISPLAY ... UPON OPERATOR-MESSAGE-DEVICE
 ACCEPT ... FROM OPERATOR-MESSAGE-DEVICE

مثال ٤ - ٨ :

SPECIAL-NAMES.
 C01 IS HEADING-AREA
 C02 IS TOTAL-AREA
 SYSIN IS FUNCTION-SELECT-INPUT
 BACK-ORDER-CODING-SEQUENCE IS STANDARD-1

لقد اختير FUNCTION- SELECT- INPUT كاسم كويل لملف المدخلات الخاص SYSIN (كلمة من كلمات IBM المحجوزة) ، يستخدم FUNCTION- SELECT- INPUT في عبارة ACCEPT من جزء الإجراءات .

ACCEPT ... FROM FUNCTION-SELECT-INPUT

مثال ٤ - ٩ :

SPECIAL-NAMES.
 SYSOUT IS ERROR-LOG.

تتجه المخرجات من عبارات DISPLAY ، من جزء الإجراءات إلى ملف مخرجات خاص بالنظام اسمه SYSOUT (كلمة من كلمات IBM المحجوزة) . اسم الكويل لـ SYSOUT يصبح ERROR- LOG أي :

DISPLAY ... UPON ERROR-LOG

٢ - ٥ قسم المدخلات والمخرجات

يبني كل تشغيل بيانات الكويل على مفهوم معالجة السجلات records في ملفات files . يتم التفكير في معلومات المدخلات (بغض النظر عن وحدة المدخلات المستخدمة) على أنها تتكون من سجلات مدخلات input records من ملف مدخلات input file . ويتم التفكير في معلومات المخرجات (بغض النظر عن وحدة المخرجات المستخدمة على أنها سجلات مخرجات out-put records وثابة تذهب إلى ملف مخرجات output file . ونظراً لمقدرة التخزين المساعد - مثل القرص على الاتصال بأي سجل في حدود الملئ الثانية (القسم الرابع من الفصل الأول) - فيمكن لهذا النوع من التخزين (يسمى بصفة عامة تخزين اتصال مباشر) أن يدعم ملفات مدخلات ومخرجات في نفس الوقت بواسطة نفس برنامج الكويل (input- output file 5) . ويمكن لسجل الاتصال المباشر أن يكون مدخلات ويجري عليه تعديلاً ثم يصبح مخرجات في نفس موقعه الأصلي على القرص ، وتسمى مثل هذه العملية التجديد في نفس الموقع update in place .

مثال ٤ - ١٠ :

يمكن أن يقوم برنامج كويل بطبع شيكات رواتب العاملين بتشغيل ٣ ملفات على الأقل .

- (١) ملف مدخلات تحتوي سجلاته (بطاقات الوقت) على حقول رقم تعريف العامل ، وعدد ساعات العمل الاسبوعية .
 - (٢) معلومات عن الراتب كمخرجات عن طريق الطابع ، تطبع على صحيفة للشيكات سابقة الإعداد . يعتبر كل سطر مطبوع سجلاً في ملف مخرجات مطبوع .
 - (٣) معلومات عن معدل الأجر لكل من العاملين يمكن أن تكون مدخلات من ملف رئيسي للعاملين على قرص . عند إدخال بطاقة الوقت لكل عامل .. يستدعى كذلك سجله من الملف الرئيسي للعاملين كمدخلات من القرص .
- وربما يراد تعديل بعض حقول الملف الرئيسي للعاملين لبعض (أو كل) العاملين ؛ نتيجة لتشغيل الرواتب مثل إضافة إجمالي الراتب للأسبوع الحالي إلى إجمالي الراتب المدفوع منذ بداية العام حتى تاريخه . يمكن أن يكون السجل الرئيسي للعامل المجدد updated مخرجات ، تسجل في نفس موقعه الأصلي على القرص (مع حذف محتوياته السابقة ، التي أصبحت متقادمة الآن) . وعلى هذا يمكن استخدام ملف القرص كملف مدخلات ومخرجات لبرنامج الرواتب ، وهذا موقف معتاد للملفات الموجودة على وحدات تخزين مساعد .

وتكوينها ... يحتوي قسم المدخلات والمخرجات على مقطع كويل واحد ، وهو مقطع التحكم في الملفات FILE- CON- TROL ، الذي يجب أن يحتوي على عبارة SELECT لكل ملف (مدخلات أو مخرجات أو مدخلات ومخرجات) خاص بالبرنامج . كل عبارة SELECT تتكون من أجزاء عديدة ، وتنتهي بنقطة (انظر الاسطر من ١٦ - ٢٢ في شكل ٤ - ١) .

اختار اسم ملف

(السطر رقم 16)

اسم الملف هو اسم يضعه المبرمج ، ويجب أن يكون وصفيًا بقدر الامكان ، مثل : EMPLOYEE- MASTER- FILE بدلا من FILE-1 . يستخدم للإشارة إلى الملف في كل عبارات البرنامج الأخرى .

مثال ٤ - ١١ :

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
  SELECT CUSTOMER-SALES-FILE ...
  SELECT SALES-REPORT ...
```

يقوم هذا البرنامج بتشغيل ملفين اثنين بالضبط (حيث إنه لا يوجد سوى عبارتي SELECT فقط) .
يشار إلى الملفين في بقية برنامج المصدر بأنهما : SALES- REPORT, CUSTOMER- SALES- FILE .

محدد الاسم الخارجى

(السطر 17)

يحدد جزء ASSIGN TO الملفات لوحدات المدخلات والمخرجات . ويعطى لكل ملف اسم خارجى، يصاحب ملف الكويل فى عبارات لغة تحكم العمل، أو فى أوامر نظام التشغيل التى تصف الملف ووحدة المدخلات أو المخرجات أكثر لنظام تشغيل الكمبيوتر (أنظر القسم رقم ٩ من الفصل الأول ، والمشكلة ٤٥ من الفصل الرابع) . يتغير شكل ومحتويات عبارات وأوامر تحكم العمل بشدة من جهاز كمبيوتر ونظام تشغيل لجهاز كمبيوتر، ونظام تشغيل آخرين وعليك أن تعرف فى هذه اللحظة كيفية كتابة الأسماء الخارجية للكمبيوتر المتاح لك استخدامه . فى كويل IBM OS/VS يجب أن يتكون الاسم الخارجى من ١ الى ٨ حروف أبجدية أو أرقام على أن يكون أول رمز حرفاً أبجدياً . ولصلحة مبرمج الكويل أن يتعلم ما يستطيع تعلمه عن لغة تحكم العمل .

مثال ٤ - ١٢ :

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
  SELECT CUSTOMER-SALES-FILE
    ASSIGN TO CUSTSALE ...
  SELECT SALES-REPORT
    ASSIGN TO SALERPT ...
```

هنا .. يتحدد الاسم الخارجى CUSTSALE ، للملف للملف CUSTOMER - SALES- FILE والاسم الخارجى
SALERPT للملف SALES- REPORT . تظهر هذه الأسماء الخارجية فى عبارات تحكم العمل، أو فى أوامر نظام التشغيل،
والتي تقدم معلومات أكثر عن وحدات المدخلات والمخرجات اللازمة لتشغيل هذه الملفات .

التنظيم تتابعى

(السطر رقم 18 : اختياري)

حالة الاتصال تتابعية

(السطر رقم 19 : اختياري)

لغة الكويل مبنية حول مفهوم تشغيل سجلات الملف سجلا سجلا . وهناك مرونة على أية حال، بغض النظر عن أى السجلات يتم تشغيله أولا وأياها ثانيا .. وهكذا . وهناك حالتان أساسيتان لتشغيل الملف أو الاتصال به وهما :

* **اتصال تتابعى sequential access** ، أو تشغيل تتابعى access processing حيث يتم تشغيل السجلات طبقا لترتيبها الواقعى على وحدة المدخلات او المخرجات، دون حاجة إلى معلومات فى الملف غير سجلات البيانات الواقعية نفسها . ويمكن أداء هذا النوع من التشغيل للقات موجودة على أى نوع من أنواع وحدات المدخلات والمخرجات .

* **اتصال عشوائى random access** أو تشغيل عشوائى random processing . يتم تشغيل السجلات طبقا للترتيب الذى يحدده البرنامج ، والذي يمكن أن يختلف عن الترتيب الذى سبق أن خزنت به السجلات واقعيا على وحدة المدخلات والمخرجات. وهذا النوع من التشغيل ممكن بالنسبة للملفات الموجودة على وحدات اتصال مباشر فقط ، وقد يتطلب أن يحتوى الملف على معلومات إضافية ، المساعدة فى تحديد سجلات البيانات التى يطلبها البرنامج .

ويشير تنظيم الملف file organization إلى الطريقة المستخدمة فى تخزين السجلات (أى معلومات إضافية تلزم للمساعدة فى تحديد موقع السجلات القريبة) فى الملف . ويقدم الكويل النمطى للمبرمج ثلاثة خيارات لتنظيم الملف .

* **تنظيم تتابعى sequential organization** . حيث يمكن وضع الملفات المرتبة تتابعيا على أى نوع من أنواع وحدات المدخلات والمخرجات . يحتوى الملف على سجلات بيانات فعلية فقط ، والتى يجب ان يتم تشغيلها بنفس الترتيب الذى وضعت فيه وحدة المدخلات والمخرجات .

* **تنظيم مفهرس indexed organization** : يجب أن توضع الملفات المفهرسة على وحدات اتصال مباشر ويمكن تشغيلها تتابعيا أو عشوائيا . عندما يتم تشغيلها تتابعيا .. تسترجع السجلات فى ترتيب تصاعدى ، طبقا لمفتاح key يسبق تعريفه (وهو حقل تعرف محتوياته كل سجل فى الملف تعريفاً فريداً مثل رقم الجزء أو رقم الحساب) . وعندما يتم تشغيلها عشوائيا .. تعرف السجلات باستخدام نفس المفتاح . ولا يحتوى الملف على سجلات البيانات الواقعية فقط لكنه يحتوى كذلك على سجلات فهرس index records التى تحتوى على معلومات خاصة بموقع كل سجل بيانات فى الملف . فإذا كان الحقل الرئيسى (حقل المفتاح) لسجل البيانات معروفاً ، فمن الممكن تحديد موقعه بسرعة مؤكدة بالبحث خلال سجلات الفهرس فى الملف ، وهذا ما يجعل التشغيل العشوائى ممكناً .

* **تنظيم نسبى relative organization** . لاتعد الملفات النسبية شائعة الاستخدام مثل الملفات التتابعية أو المفهرسة . يحتوى الملف على سجلات بيانات فعلية فقط ، ترقم بالارقام ١ و ٢ و ٣ و... ويرقم السجل record number .. هذا يمكن تشغيل الملف عشوائيا .

وحيث أن مجالات الأعمال معتادة على تعريف المعلومات المحفوظة في سجلات بواسطة مفتاح ket ، وليس رقم سجل record number .. فلاتستخدم الملفات النسبية إلا عندما يكون الاتصال السريع جدا ضروريا ؛ وإذا يجب أن توضع الملفات النسبية على وحدات اتصال مباشر .

توفر عبارة SELECT وسيلة لمبرمج الكوبل لاختيار تنظيم الملف وحالة الاتصال لكل ملف يراد تشغيله بواسطة البرنامج. هذا الخيار يكون حاسما . عادة ماتنتج الملفات التي تتطلب تشغيللا عشوائيا بتنظيم مفرس ، أما الملفات التي تتطلب تشغيللا تتابعيا فقط .. فتنتج بتنظيم تتابعي ، وتنتج الملفات التي تتطلب كلاً من التشغيل التتابعي والتشغيل العشوائي بتنظيم مفرس (الذي يوفر كلاً من نوعي التشغيل) . إن تشغيل الملفات تتابعيا مشروح بالتفصيل في الفصل الحادي عشر ، أما الملفات المفهرسة والنسبية فتقع خارج نطاق هذا الملخص .

مثال ٤ - ١٣ :

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
  SELECT CUSTOMER-SALES-FILE
    ASSIGN TO CUSTSALE
    ORGANIZATION IS SEQUENTIAL
    ACCESS IS SEQUENTIAL...
  SELECT SALES-REPORT
    ASSIGN TO SALERPT
    ORGANIZATION IS SEQUENTIAL
    ACCESS IS SEQUENTIAL...
```

هناك عبارتا SELECT اللتين :

CUSTOMER- SALES- FILE (والاسم الخارجى CUSTSALE)

و SALES- REPORT (والاسم الخارجى SALERPT) .

• ORGANIZATION IS SEQUENTIAL

يمكن أن يحذف هذا الجزء نظرا لأن التنظيم التتابعى للملفات هو التقليدى ، وهناك إمكانيات أخرى، وهى :

• ORGANIZATION IS RELATIVE, ORGANIZATION IS INDEXED

• ACCESS MODE IS SEQUENTIAL

يمكن حذف هذا الجزء كذلك لان الاتصال التتابعى هو التقليدى ، وهناك إمكانيات أخرى وهى :

ACCESS MODE IS RANDOM و ACCESS MODE IS DYNAMIC

الملفات المرتبة تتابعيا يمكن الاتصال بها تتابعيا فقط ، وتتطلب حالات الاتصال الأخرى تنظيما مفرسا أو تنظيما نسبيا .

حالة الملف هي

(السطر رقم 20 : اختياري)

من ناحية المفهوم.. فمعالجة جزء الإجراءات لسجلات الملف في الكويل تكون بسيطة جدا ؛ إذ تبدأ عبارة READ بسجل واحد ، بإبخاله من ملف مدخلات من وحدة مدخلات أو وحدة تخزين مساعد في ذاكرة وحدة التشغيل المركزية للتشغيل ، وتنقل عبارة WRITE سجلا من ذاكرة وحدة التشغيل المركزية الى وحدة مخرجات أو وحدة تخزين مساعد، يكون بها ملف المخرجات . وقبل ان يمكن استخدام عبارة READ ، أو عبارة WRITE في تشغيل السجلات الموجودة في ملف .. يجب أن تستخدم عبارة OPEN لاعداد الملف للتشغيل (انظر مثال ٢ - ٥) . وبالمثل يجب استخدام عبارة CLOSE لإنهاء نشاط الملف بعد الانتهاء من تنفيذ كل عبارات WRITE, READ .

وفي الحياة الواقعية .. فإنه من السهل على أية حال حدوث أخطاء للأشياء . فمثلا قد تفشل عملية OPEN نظرا لعدم توافق وصف الملف في برنامج مصدر الكويل، مع الوصف المصاحب في عبارات تحكم العمل المصاحبة . ويمكن ان تفشل عملية WRITE في اضافة سجل إلى ملف ، مرتب ترتيباً تتابعياً على قرص، وذلك لسبب عدم وجود مكان متاح على القرص. وفي الواقع .. فإن توقع فشل معين يكون مبنيا داخل برامج الكويل. وعلى هذا .. لتشغيل السجلات في ملف تتابعي.. فإننا نصمم على ذلك البرنامج الذي يحفظ قراءة سجل، ويشغل السجل، ويخرج النتائج حتى تفشل READ بسبب الانتهاء من قراءة كل سجلات الملف (وهذا يسمى شرط نهاية الملف) .

يجب أن تكتب البرامج لمعالجة الشروط الاستثنائية exception conditions التي يمكن ان تحدث أثناء التشغيل . وتوفر عبارة SELECT وسيلة للحصول على تغذية مرتجعة feedback ، خاصة بنجاح أو فشل كل عملية يتم إجراؤها على الملف . وهذه التغذية المرتجعة تقدم على هيئة منطقة بيانات ينقل فيها شفرة معينة بعد كل عملية من عمليات WRITE, READ, CLOSE, OPEN تجرى على الملف . وتذكر الشفرة ما إذا كانت العبارة ناجحة أم لا - فإذا لم تكن ناجحة فما هو السبب .

مثال ٤ - ١٤ :

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
  SELECT MONTHLY-SALES-FILE
  ASSIGN TO MSALES
  FILE STATUS IS MONTHLY-SALES-FILE-STATUS
```

في جزء FILE STATUS أختار المبرمج اسم MONTHLY- SALES- FILE- STATUS لمنطقة البيانات التي تستخدم في توفير التغذية المرتجعة الخاصة بنتائج كل عملية من عمليات جزء الإجراءات (OPEN, WRITE, READ, CLOSE) التي تم تنفيذها على الملف . يجب أن يعرف MONTHLY- SALES- FILE- STATUS في جزء البيانات كمجموعة يمكن أن تحتفظ بشفرة من خانتين تسجل نجاح أو فشل أي عملية معطاه .. على الملف . أفحص دليل المورد لمعنى FILE STATUS ، الخاص بالنظام المتاح لك استخدامه .

لا يكلف الكثير من مبرمجي الكويل ذوي الخبرة أنفسهم تعريف منطقة FILE STATUS بالنسبة للملفات المرتبة تتابعيا ، حيث إن هناك طرقاً أخرى للتأكد من اتمام عمليات المدخلات والمخرجات بنجاح . (انظر الفصل الحادي عشر) .

جزء مناطق

(السطر رقم 21 : اختياري)

يمكن استخدام هذا الجزء للإسراع من تشغيل الملف عندما يكون متابعيا (انظر الذاكرات المتعددة - ٥ - ٨)

٤ - ٦ أمثلة على جزء الأوساط

مثال ٤ - ١٥

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-370.
OBJECT-COMPUTER. IBM-370.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT OLD-INVENTORY-MASTER-FILE
    ASSIGN TO OLDMAST.
    SELECT NEW-INVENTORY-MASTER-FILE
    ASSIGN TO NEWMAST.
    SELECT TRANSACTION-FILE
    ASSIGN TO TRANS.
```

هذا جزء أوساط لبرنامج تجديد ملف متابعي يستخدم سجلات ملف العمليات الجارية TRANSACTION FILE في تجديد المعلومات ، في سجلات ملف المخزون الرئيسي القديم OLD- INVENTORY- MASTER- FILE . وتوضع السجلات التي تحتوي على التجديد في ملف المخزون الرئيسي الجديد NEW- INVENTORY- MASTER- FILE . كل الملفات الثلاثة متتابعة (الشيء التقليدي عندما يكون جزء ORGANIZATION IS محذوفا) ، ويتم تشغيلها متابعيا (الشيء التقليدي عندما يكون جزء ACCESS MODE IS محذوفا) .

وحيث إن كل الملفات متتابعة ، فإن مناطق شفرة FILE STATUS تكون اختيارية ولا تستخدم (يجرى اختبار الشروط الإستثنائية أثناء عمليات المدخلات والمخرجات باستخدام طرق جزء الإجراءات الأخرى) .

تسلسل المتابع يكون تقليديا لنظام الكمبيوتر EBCDIC لنظام IBM 370 . وحيث انه لم يتحدد حالة تصحيح DE-BUGGING MODE ، تعامل الاسطر الموجود حرف D في عمودها السابع كتعليقات ولا يكون لها أى تأثير على تنفيذ البرنامج . الأسماء ASSIGN TO الشكل الصحيح بالنسبة لنظم IBM OS/VS (TRANS, NEWMAST, OLDMAST) تحتوي على رموز تتراوح من ١ - ٨ وهي حروف أو أرقام فقط وأول رمز حرف) .

مثال ٤ - ١٦ :

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-370
    WITH DEBUGGING MODE.
OBJECT-COMPUTER. IBM-370.
SPECIAL-NAMES.
    C01 IS TOP-OF-CHECK
    C02 IS PAYEE-AREA
    C05 IS AMOUNT-AREA

INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT PAYROLL-FILE
        ASSIGN TO PAYROLL
        ORGANIZATION IS SEQUENTIAL
        ACCESS MODE IS SEQUENTIAL
        FILE STATUS IS PAYROLL-STATUS-CODE

    SELECT PAYCHECK-FILE
        ASSIGN TO CHECKS
        ORGANIZATION IS SEQUENTIAL
        ACCESS MODE IS SEQUENTIAL
        FILE STATUS IS CHECK-STATUS-CODE
```

تعليقات :

- (١) محدد حالة تصحيح DEBUGGING MODE ، وعلى هذا تترجم الاسطر الموجود حرف D فى عمودها السابع ويكون لها تأثير أثناء التنفيذ.
- (٢) لاحظ استخدام النقاط المرتبة فى إنهاء الاسماء الخاصة وعبارة الاختيار SELECT.
- (٣) القنوات رقم ١ ، ٢ ، ٣ لآلية التحكم فى عربة الطباعة، تستخدم فى وضع الطابع (نقل الطابع) إلى الثلاثة مناطق من الشيكات المراد طباعتها بواسطة البرنامج .
- (٤) يستخدم البرنامج ملفين . باستخدام معلومات المدخلات من PAYROLL- FILE تتم طباعة الشيكات فى ملف PAYCHECK- FILE . للاسماء الخارجية للملفات (CHEKS & PAYROLL) نفس التشكيل المطلوب من نظم IBM OS/VS
- (0) جزئى ORGANIZATION & ACCESS MODE محددان، بالرغم من أن لهما القيمة التقليدية SEQUEN- TIAL. وقد تم عمل ذلك لتقديم توثيق واضح للبرنامج .
- (٦) صاحب كل من الملفين مناطق شفرة FILE STATUS ، التى يمكن اختبارها بعد كل عملية على ملف فى جزء الإجراءات. يجب أن يزداد تعريف كل من CHECK- STATUS- CODE, PAYRPLL- STATUS- CODE فى جزء البيانات . وكان من الممكن حذف مناطق FILE STATUS ومعالجة شروط الاستثناء بطرق أخرى (انظر الفصل الحادى عشر).

استئلة مراجعة

- ٤ - ١ صف تكوين جزء الأوساط .
- ٤ - ٢ ما الغرض من قسم التشكيل ؟
- ٤ - ٣ ما الغرض من قسم المدخلات والمخرجات ؟
- ٤ - ٤ اذكر قواعد كتابة المدخلات والمخرجات .
- ٤ - ٥ وضع الغرض من اسطر التصحيح ، وكيف تتأثر بمقطع كمبيوتر المصدر ؟
- ٤ - ٦ ما معنى تسلسل التتابع ؟
- ٤ - ٧ كيف يفسر تسلسل التتابع التقليدى ؟
- ٤ - ٨ ماذا يجب عمله ليتمكن استخدام تسلسل تتابع غير تقليدى ؟
- ٤ - ٩ ما الوظائف الثلاث التى يؤديها مقطع الاسماء الخاصة ؟
- ٤ - ١٠ ما المقصود بالصيغ المستمرة ؟
- ٤ - ١١ ما المقصود بشريط تحكم عربية الطباعة ؟
- ٤ - ١٢ وضع كيف تربط قنوات تحكم العربية بمناطق الصيغ المستمرة .
- ٤ - ١٣ ماذا تستخدم بعض الطابعات الحديثة بدلا من شريط تحكم عربية الطباعة ؟
- ٤ - ١٤ ما تسلسل التتابع المعروف بانه STANDARD-1 وما تسلسل التتابع المعروف بانه NATIVE فى نظم IBM ؟
- ٤ - ١٥ ما النهاية الطرفية لمشغل الكمبيوتر الأسمى ؟
- ٤ - ١٦ ماذا تفعل عبارات DISPLAY, ACCEPT من جزء الإجراءات ؟ كيف ترتبط هذه العبارات بمقطع الاسماء الخاصة من جزء الأوساط ؟
- ٤ - ١٧ اذكر معنى الاسماء الوظيفية التالية فى كويل IBM OS/VS
C04 (هـ) C01 (د) SYSOUT (جـ) CONSOLE (ب) SYSIN
- ٤ - ١٨ ما هى وحدات الاتصال المباشر
- ٤ - ١٩ ماهى عملية التجديد فى نفس الموقع ؟
- ٤ - ٢٠ اذكر بعض الامثلة للسجلات والملفات .
- ٤ - ٢١ صف كل جزء من اجزاء عبارة SELECT و اشرح مايفعله .
- ٤ - ٢٢ ما هى لغة تحكم العمل JCL ؟
- ٤ - ٢٣ ما هى لغة اوامر نظام التشغيل ؟
- ٤ - ٢٤ كيف يرتبط جزء ASSIGN TO من عبارة SELECT بلغة تحكم العمل ؟
- ٤ - ٢٥ وضع اتصال الملف تتابعيا (أو تشغيل الملف تتابعيا) ؟
- ٤ - ٢٦ وضع اتصال الملف عشوائيا (أو تشغيل الملف عشوائيا) .
- ٤ - ٢٧ عرف تنظيم الملف ؟
- ٤ - ٢٨ وضع بإيجاز ترتيب (تنظيم) الملف تتابعيا .
- ٤ - ٢٩ وضع بإيجاز ترتيب (تنظيم) الملف المفهرس .
- ٤ - ٣٠ وضع بإيجاز ترتيب (تنظيم) الملف النسبى .

- ٤ - ٣١ لماذا يعتبر اختيار تنظيم الملف مهما ؟
 ٤ - ٣٢ اربط الحاجة الى اتصال تنابعي والحاجة الى اتصال عشوائي باختيار تنظيم الملف ؟
 ٤ - ٣٣ وضع بإيجاز الغرض من عبارات جزء الإجراءات التالية :
 CLOSE, WRITE, READ, OPEN
 ٤ - ٣٤ اذكر بعض شروط الاستثناءات التي يمكن أن تحدث أثناء تشغيل الملف .
 ٤ - ٣٥ كيف يشير جزء FILE STATUS من عبارات SELECT الى معالجة شروط استثناءات ؟
 ٤ - ٣٦ ما هو شرط نهاية الملف ؟

مسائل محلولة

- ٤ - ٣٧ أكتب جزء لوساط البرنامج يختبر على نظام كمبيوتر IBM 370 . اسطر التصحيح سوف تترجم الى برنامج الهدف ويستخدم تسلسل التتابع EBCDIC .

CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-370 WITH DEBUGGING MODE.
 OBJECT-COMPUTER. IBM-370.

أو
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-370
 WITH DEBUGGING MODE.
 OBJECT-COMPUTER. IBM-370
 PROGRAM COLLATING SEQUENCE IS EBCDIC-ORDER.
 SPECIAL-NAMES.
 EBCDIC-ORDER IS NATIVE.

- ٤ - ٣٨ اذكر كل الأشكال المختلفة الصحيحة للسطر الخامس، في شكل (٤ - ١)

(1) PROGRAM COLLATING SEQUENCE IS alphabet-name; (2) PROGRAM SEQUENCE IS alphabet-name; (3) PROGRAM SEQUENCE alphabet-name; (4) COLLATING SEQUENCE IS alphabet-name; (5) SEQUENCE IS alphabet-name; (6) PROGRAM COLLATING SEQUENCE alphabet-name; (7) COLLATING SEQUENCE alphabet-name; (8) SEQUENCE alphabet-name; (9) blank line.

- ٤ - ٣٩ يراد طباعة شيكات بواسطة الكمبيوتر . وقد اعدت قنوات تحكم العربية لتتفق مع مناطق موجودة على الشيك على النحو التالي : القناة ١ للتاريخ date القناة ٢ للمستفيد من الشيك payee والقناة ٥ للكمية amount . ويعرض البرنامج على نهاية مشغل الكمبيوتر الطرفية إجمالي عدد الشيكات المطبوعة كذلك . اكتب مقطع الاسماء الخاصة لهذا البرنامج.

SPECIAL-NAMES.
 C01 IS CHECK-DATE-AREA
 C02 IS CHECK-PAYEE-AREA
 C05 IS CHECK-AMOUNT-AREA
 CONSOLE IS CHECK-CONTROL-COUNT-DEVICE

٤ - ٤٠ اكتب قسم التشكيل لبرنامج مستخدما تسلسل التتابع التالي :

B بعدها J ثم A , K , ..., W , C , I , 7 , 0 , 1 , 2 , ... 6 , 9 .

تم تصحيح ، البرنامج تماما ويراد ترجمته وتنفيذه على كمبيوتر من طراز IBM 370

CONFIGURATION SECTION.

SOURCE-COMPUTER. IBM-370.

OBJECT-COMPUTER. IBM-370

COLLATING SEQUENCE IS
WAREHOUSE-BIN-ORDER.

SPECIAL-NAMES.

WAREHOUSE-BIN-ORDER IS

"B" "J" "A"
"K" THRU "W"
"C" THRU "I"
"7" "1" "0"
"2" THRU "6"
"9"

٤ - ٤١ اكتب قسم التشكيل لبرنامج ينفذ على كمبيوتر من طراز IBM 370 ، وله تسلسل تتابع ASCII ، بدلا من تسلسل التتابع التقليدي EBCDIC ، ويجب أن تترجم أسطر التصحيح .

CONFIGURATION SECTION.

SOURCE-COMPUTER. IBM-370 DEBUGGING MODE.

OBJECT-COMPUTER. IBM-370 SEQUENCE ASCII-ORDER.

SPECIAL-NAMES.

ASCII-ORDER IS STANDARD-1.

٤ - ٤٢ عادة ماتسمى قارئات البطاقات ، والطابعات ، ومثقيات البطاقات بوحدة السجل unit record devices .
اشرح مفاهيم الملف والسجل والحقل بالنسبة الى (أ) مجموعة بطاقات . (ب) تقرير تحليل مبيعات مطبوع على صيغ مستمرة .

(أ) كل بطاقة عبارة عن سجل ، ومجموعة البطاقات هي الملف ، وكل قطعة معلومات منفصلة مثقبة على البطاقة هي حقل ، وذلك مثل رقم البطاقة أو الكمية المطلوبة .
(ب) مجموعة كل الاسطر المطبوعة في التقرير هي الملف ، وكل سطر عبارة عن سجل ، وكل قطعة معلومات على السطر عبارة عن حقل (بما في ذلك الفراغات الفاصلة) .

٤ - ٤٣ صف كيف يمكن عمل تجديد سجل قرص في موقعه ، بالنسبة إلى عمليات حسابات توفير البنك .

اعتبر أن هناك بنكاً له نهايات طرفية للصرف في وسط الخط المفتوح (أى أن النهايات الطرفية متصلة بنظام كمبيوتر البنك، أو بواسطة كابل كهربائى أو بواسطة اتصالات برقية) . إذا أراد أحد العملاء عمل إيداع أو سحب من حساب التوفير، فيكتب الصراف مستخدماً النهاية الطرفية رقم الحساب ونوع العملية الجارية والكمية المتعامل بها .

وهذه المعلومات هي مدخلات بواسطة برنامج كمبيوتر، يتصل عشوائياً بسجل ملف قرص به موازنة حساب التوفير لهذا العميل (للسماح بالتحصيل العشوائى يمكن أن يكون الملف مفهرساً طبقاً لأرقام حسابات التوفير) . يجدد عند ذلك البرنامج موازنة الحساب بإضافة أو طرح كمية العملية الجارية . وأخيراً يكتب سجل العميل المجدد في موقعه الاصل على القرص .

٤ - ٤٤ حدد أسماء ملفات الكوبل التالية ، وما إذا كانت صحيحة أم ضعيفة، أم غير صحيحة .

- (a) SELECT REPORT ASSIGN TO ...
- (b) SELECT REPORT-FILE ASSIGN TO ...
- (c) SELECT OUT-OF-STOCK-ITEMS-REPORT ASSIGN TO ...
- (d) SELECT INPUT-FILE ASSIGN TO ...

(ا) غير صحيح : كلمة REPORT هي كلمة محجوزة .

(ب) صحيح لكنه ضعيف : REPORT- FILE ليس وصفيًا .

(جـ) صحيح .

(د) صحيح لكنه ضعيف : ربما يكون هناك عديد من ملفات المدخلات .

٤ - ٤٥ يجب أن يوصف كل ملف يستخدم في برنامج كوبل بعبارة SELECT في قسم المدخلات والمخرجات . ما الغرض من ASSIGN TO في عبارة SELECT ؟

جزء الأوساط (حيث أنه يتعامل مع أوساط نظم مكونات البرنامج) هو الجزء الأكثر احتمالاً للتغيير من برنامج الكوبل إذا غيرت الكمبيوتر المستخدم .

ونعطي هنا مثالين :

(أ) يتطلب كوبل IBM OS/VS (لنظم IBM 370) .. إنه يجب أن يوصف كل ملف وصفاً إضافياً في عبارة تحكم عمل، تسمى تعريف بيانات (data- definition) DD . واسم ASSIGN TO هو الواصل المشترك بين عبارة DD والمعلومات في برنامج مصدر الكوبل . وعلى هذا فالعبارة :

SELECT INVENTORY- MASTER ASSIGN TO INVMASST..

يمكن أن تكون عبارة DD المصاحبة على النحو التالى :

توضيحها

عبارة DD

INVMASST DD UNIT= DISK, // الوحدة الفعلية هي قرص

VOL= SER= ABC123, // مجموعة الأقراص اسمها ABC123,

DISP= OLD, // الملف موجود فعلاً

DSNAME= INVNTY. MASTER. A17 // اسم الملف في عنوان الملف

وجود ASSIGN TO وهو INVMAS7 فى بداية عبارة DD يكون إجباريا .

(ب) يستخدم كويل Burroughs (لنظم كمبيوتر B1910) لغة أوامر نظام تشغيل بدلا من لغة تحكم العمل . ويستطيع المستفيد الموجود أمام نهاية طرفية أن يكتب برنامج مصدر كويل، ثم يكتب عبارة لغة أوامر تتسبب فى ترجمة البرنامج .

افرض أن برنامج المصدر به عبارة SELECT التالية :

SELECT INVENTORY ASSIGN TO DISK

اسماء ملفات B1910 محددة بعدد عشرة رموز (INVENTORY) . DISK كاسم فى ASSIGN TO وهى كلمة محجوزة تحدد للنظام أن INVENTORY موجود على وحدة قرص مغناطيسى . وفى كويل B1910 .. يخدم اسم الملف IN- VENTORY كواصل بين تعريف الملف فى البرنامج، وعبارات لغة الأوامر التى تعرف أكثر . فإذا كان تعريف البرنامج هو INVENTORY فيمكن أن يكتب الأمر التالى لتنفيذ البرنامج المترجم :

EXECUTE INVUPDATE, FILE INVENTORY NAME INVMAS7

يطلب امر EXECUTE من نظام التشغيل ان ينفذ برنامج الهدف المحدد . ويجعل جزء FILE الملف المسمى INVENTO- RY على القرص . وعندما ينفذ البرنامج .. يتجدد ملف القرص INVMAS7.

٤ - ٤٦ حدد اختياراتك لتنظيم الملفات، وحالات الاتصال بالنسبة للتطبيقات التالية :

(أ) ملف رئيسى لحد المديونية لشركة تسمح للتجار الآخرين بالاتصال ماتقيا لاسترجاع حد مديونية بطاقات ائتمان العملاء .

(ب) ملف شريط لاسماء وعناوين تستخدم فى طباعة العناوين البريدية .

(ج) ملف قرص لحسابات المدينين به سجلات بموازانات العملاء الحالية .

(أ) التنظيم : فهرس (طبقا لرقم بطاقة الائتمان) الاتصال عشوائى .

(ب) التنظيم : تتابعى (الخيار الوحيد لوحدة الاتصال غير المباشر) . الاتصال : تتابعى (الاختيار الوحيد للشريط).

(ج) التنظيم : إما فهرس (طبقا لرقم العميل) أو تتابعى (مرتب طبقا لرقم العميل) . الاتصال : يجب أن يجدد الملف بمعلومات عن المدفوعات والمشتريات الجديدة . فإذا كان التنظيم فهرسا ، فيمكن أن يحدث هذا عشوائيا . أما إذا كان ملف المدفوعات والمشتريات مرتبا كذلك طبقا لرقم العميل ، يمكن أن يتحقق التجديد بالاتصال العشوائى التتابعى . ويتم طباعة التقارير (مثل الموازنات القديمة) باستخدام الاتصال التتابعى .

٤ - ٤٧ ما المقصود بشرط نهاية الملف ؟

يطبق مفهوم شرط نهاية الملف عندما يتم تشغيل الملف تتابعيا فى المدخلات . يقرأ أول سجل فى الملف أولا، ويليه السجل الثانى فالثالث... وهكذا . فإذا كان فى الملف 7000 سجل فتحدث نهاية الملف عندما تحدث محاولة القراءة 7001 وتقتل .

٤ - ٤٨ اكتب جزء أوساط كاملاً لبرنامج لطباعة شيكات . يستخدم البرنامج ملف شريط للمدخلات به سجلات الوقت الاسبوعية مرتباً طبقاً لرقم العامل ، ويستخدم ملف عاملين رئيسياً على قرص كمدخلات كذلك ، مرتباً ، وطبقاً لرقم العامل ، كل تشغيل الملفات يحدث تتابعياً ، والبرنامج سبق تصحيحه . لا تستخدم قنوات تحكم العربية في هذا البرنامج ، وذلك بالرغم من أن بعض الرسائل يمكن أن تظهر في النهاية الطرفية لمشغل الكمبيوتر.

ENVIRONMENT DIVISION.

CONFIGURATION SECTION.

SOURCE-COMPUTER. IBM-370.

OBJECT-COMPUTER. IBM-370.

SPECIAL-NAMES.

CONSOLE IS OPERATOR-MESSAGES-DEVICE

INPUT-OUTPUT SECTION.

FILE-CONTROL.

SELECT WEEKLY-TIME-FILE
ASSIGN TO WKLYTIME
ORGANIZATION IS SEQUENTIAL
ACCESS IS SEQUENTIAL
FILE STATUS IS TIME-FILE-STATUS

SELECT EMPLOYEE-MASTER-FILE
ASSIGN TO EMPMAST
ORGANIZATION IS SEQUENTIAL
ACCESS IS SEQUENTIAL

SELECT PAYCHECK-FILE
ASSIGN TO CHECKS
ORGANIZATION IS SEQUENTIAL
ACCESS IS SEQUENTIAL

(هل نسيت أن الشيكات المطبوعة تكون ملفاً ، وتتطلب على هذا عبارة SELECT ؟ ملاحظة : (١) الاسطر الفارغة والنقاط المرتبة . (٢) حذف DEBUGGING MODE و COLLATING SEQUENCE . (٣) مصاحبة اسم الكويل TIME- FILE- STATUS مع OPERATOR- MESSAGE- DEVICE للنظام (٤) حددت منطقة اسمها WEEKLY- TIME- FILE ، وعرفت أكثر في جزء البيانات... حيث يوضع فيها شفرة تغذية مرتجعة بعد كل عملية تجرى على الملف .

نهايين بوجبة

من ٤٩ إلى ٥٨ عدل التمارين من ٢٩ - ٣٨ في الفصل الثاني لتعكس ماتعلمته عن جزء الأوساط فيها .

الفصل الخامس جزء البيانات Data Division

٥ - ١ هيكل جزء البيانات

يحدد شكل ٥ . ١ الهيكل العريض لجزء البيانات ، والذي سبق وصف وظيفته العامة في القسم الأول من الفصل الثاني .

```
DATA DIVISION.  
[ FILE SECTION.  
  [file description entry  
  record description entry] ... ]  
[ WORKING-STORAGE SECTION.  
  [data item or "record" description entry] ... ]
```

شكل (٥ - ١)

مع أن كلا من قسم الملفات FILE SECTION ، وقسم مخزن العمل WORKING- STORAGE SECTION اختياري، إلا إنهما مطلوبان بصفة عامة عملياً ، حيث يستخدم كل برنامج تقريباً ملفاً واحداً على الأقل وبعض عدادات ، وشعارات ، ومناطق عمل في مخزن العمل . ويوضح مثال ٥ - ١ جزء بيانات فعلياً منظرًا لجزء الأوساط مثال ٤ - ١ .

مثال ٥ - ١ :

```
00066 DATA DIVISION.  
00067  
00068 FILE SECTION.  
00069  
00070 FD CUSTOMER-SALES-FILE  
00071 BLOCK CONTAINS 0 RECORDS  
00072 RECORD CONTAINS 80 CHARACTERS  
00073 LABEL RECORDS ARE STANDARD  
00074 .  
00075  
00076 01 CUSTOMER-SALES-RECORD.  
00077 05 CUST-SALES-CUSTOMER-NO PIC X(4).  
00078 05 CUST-SALES-SALES-ID PIC X(6).  
00079 05 CUST-SALES-AMOUNT PIC S9(7)V99.  
00080 05 FILLER PIC X(61).
```

```

00081      FD  SALES-REPORT
00082          RECORD CONTAINS 132 CHARACTERS
00083          LABEL RECORDS ARE OMITTED
00084          .
00085
00086
00087      01  SALES-REPORT-LINE                      PIC X(132).
00088
00089      WORKING-STORAGE SECTION.
00090
00091      01  END-OF-FILE-AND-STATUS-AREA.
00092
00093          05  CUSTOMER-SALES-FILE-END          PIC X.
00094              88  NO-MORE-CUSTOMER-RECORDS      VALUE "T".
00095          05  CUST-SALES-FILE-STATUS          PIC XX.
00096          05  SALES-REPORT-STATUS             PIC XX.
00097
00098
00099
00100      01  TOTAL-AND-COMPARISON-AREAS.
00101          05  OLD-CUSTOMER-NO                  PIC X(4).
00102          05  CUSTOMER-TOTAL                   PIC S9(9)V99  COMP-3.
00103          05  GRAND-TOTAL                     PIC S9(9)V99  COMP-3.
00104
00105      01  COUNTER-AREA.
00106          05  LINE-COUNT                      PIC S9(3)      COMP-3.
00107          05  PAGE-COUNT                     PIC S9(5)      COMP-3.
00108          05  LINE-SKIP                      PIC S9(3)      COMP-3.
00109
00110      01  HEADING-LINE.
00111          05  FILLER                          PIC X(10)      VALUE SPACES.
00112          05  FILLER                          PIC X(20)
00113              VALUE "SALES BY CUSTOMER".
00114          05  FILLER                          PIC X(10)      VALUE SPACES.
00115          05  OUTPUT-PAGE-COUNT              PIC ZZ,ZZ9.
00116          05  FILLER                          PIC X(86)      VALUE SPACES.
00117
00118      01  ERROR-LINE.
00119          05  ERROR-CUSTOMER-NO              PIC X(4).
00120          05  FILLER                          PIC X(2)      VALUE SPACES.
00121          05  FILLER                          PIC X(22)
00122              VALUE "*** OUT OF SEQUENCE ***".
00123          05  FILLER                          PIC X(104)     VALUE SPACES.
00124
00125      01  OUTPUT-TOTAL-LINE.
00126          05  FILLER                          PIC X(30)      VALUE SPACES.
00127          05  OUTPUT-CUSTOMER-TOTAL          PIC $$$$,$$$,$$$.$99-.
00128          05  FILLER                          PIC XX        VALUE " *".
00129          05  GRAND-TOTAL-FLAG              PIC X.
00130          05  FILLER                          PIC X(83)      VALUE SPACES.
00131
00132      01  OUTPUT-LINE.
00133          05  OUTPUT-CUSTOMER-NO            PIC X(4).
00134          05  FILLER                          PIC X(10)      VALUE SPACES.
00135          05  OUTPUT-SALES-ID               PIC X(6).
00136          05  FILLER                          PIC X(10)      VALUE SPACES.
00137          05  OUTPUT-AMOUNT                 PIC $$$,$$$,$$$.$99-.
00138          05  FILLER                          PIC X(88)      VALUE SPACES.
00139
00140
00141
00142

```


٥ - ٢ قسم الملفات

يجب أن يحتوى قسم الملفات على جزء وصف الملف (FD) file descriptiptin entry لكل ملف، له عبارة SELECT فى جزء الأوساط .

مثال ۵ - ۲ :

```
line 52: SELECT CUSTOMER-SALES-FILE
line 59: SELECT SALES-REPORT
```

يسمى الملفان المستخدمان في البرنامج ، وتبدأ محتويات FD المناظرة في مثال ٥ - ١ على الأسطر (٧٠ - ٨٢) .
بعد معلومات FD .. يجب أن يوجد لكل ملف مستوى 01 واحد على الأقل خاص بوصف السجل - record description en-
try يصف تخطيط السجلات في هذا الملف ، والهيكل التكويني لمحتوى FD في شكل ٥ - ٢ .

```

FD file-name
[ BLOCK CONTAINS [integer-1 TO] integer-2 { CHARACTERS
RECORDS }
[RECORD CONTAINS [integer-3 TO] integer-4 CHARACTERS]
LABEL { RECORD IS } { STANDARD }
{ RECORDS ARE } { OMITTED }
[ DATA { RECORD IS } data-name-3 [data-name-4] ... ]
[ LINAGE IS { data-name-5 } integer-5 LINES
[ WITH FOOTING AT { data-name-6 } integer-6 ] ]
LINES AT TOP { data-name-7 } integer-7
LINES AT BOTTOM { data-name-8 } integer-8 ]

```

شکل (۵ - ۲)

يجب أن يبدأ كتابة عنوان وصف الملف "FD" في المنطقة A ، أما بقية الأجزاء فتكتب في المنطقة B . اسم الملف (وهو نفس الاسم المستخدم في عبارة SELECT) ، يجب أن يظهر أولاً ، بعد FD مباشرة . ويمكن ظهور بقية الأجزاء بأى ترتيب . يجب استخدام الأسطر الفارغة (أو أسطر التعليقات الفارغة) والترحيل لتسهيل القراءة . ولا توجد الا نقطة واحدة فقط ، موضوعة في نهاية محتويات FD (من الأفضل جعلها نقطة مرتبة) .

DATA DIVISION.

مثال ۵ - ۳ :

FILE SECTION.

FD WEEKLY-PERSONNEL-REPORT

BLOCK CONTAINS 1 RECORD
RECORD CONTAINS 132 CHARACTERS
LABEL RECORDS ARE OMITTED
DATA RECORD IS PERSONNEL-REPORT-RECORD
LINAGE IS 50 LINES
WITH FOOTING AT 45
LINES AT TOP 8
LINES AT BOTTOM 8

01 PERSONNEL-REPORT-RECORD...

لاحظ استخدام الأسطر الفارغة والترحيل والنقطة المرتبة . أجزاء وصف السجل التي تتبع FD بوصف كل حقل من حقول سجلات الملف . يجب أن يكون اسم FD وهو WEEKLY- PERSONAL- REPORT نفس الاسم المستخدم في عبارة SELECT للملف .

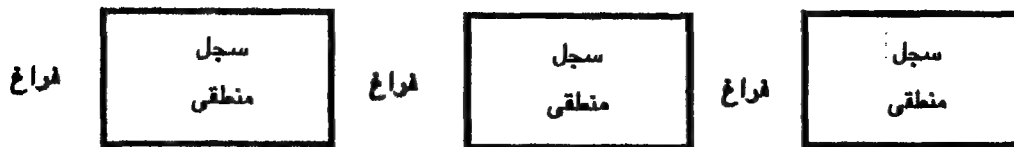
في الأقسام من ٢ إلى ٧ من هذا الفصل.. نناقش أجزاء FD كما هو مبين في شكل ٥ - ٢ جزءاً جزءاً . تذكر أن الغرض من كل جزء هو تقديم معلومات خاصة بكيفية تخزين سجلات الملف في الوحدات الواقعية .

٥ - ٣ وصف الملف وما تحتويه المجموعة

عادة ماتجمع blocked الملفات الموجودة على أقراص، أو شرائط على هيئة مجموعات، لتحقيق استغلال أكثر كفاءة للمكان والإسراع من تشغيل الملف . وفي عملية التجميع .. ميز بين السجل المنطقي logical record (الذي يجرى تشغيله لواحد فقط منه، في نفس الوقت بواسطة البرنامج)، أو السجل الواقعي physical (وحدة المعلومات التي تنقل من وإلى وحدة المدخلات والمخرجات) .

مثال ٥ - ٤ :

الملف الذي لا يحتوي على مجموعات الموجودة على شريط أو قرص يحتوي على سجلات منطقية ، منفصلة عن بعضها البعض بفراغات ما بين السجلات interrecord gaps :



أما في السجلات المحتوية على مجموعات .. فتجمع السجلات المنطقية على هيئة سجلات واقعية، منفصلة عن بعضها البعض بفراغات ما بين المجموعات interblock gaps .



سجل واقعي

في هذا الملف تجمع فيه السجلات أربعة سجلات منطقية في كل سجل واقعي .. ونقول على هذا إن له معامل تجميع blocking factor (BF) يساوي ٤ ، وبالنسبة إلى الملف غير المجمع فإن معامل التجميع يساوي ١ . ولأي ملف تقرأ أو تكتب كل الرموز الموجودة بين فراغين متتاليين كوحدة واحدة . وعلى هذا .. يتطلب الملف المجمع - في التشغيل التتابعي - $1/4$ عمليات المدخلات والمخرجات فقط التي يتطلبها الملف غير المجمع . وينخفض إجمالي وقت وضع القرص أو الشريط إلى حوالي $1/4$ الوقت المخصص للملف المجمع . وحيث أن الملف المجمع $1/4$ الفراغات فقط .. فالمكان غير المشغول منه يقل عن الملف غير المجمع .

مثال ٥ - ٥ :

افترض إنه مطلوبه عمل تشغيل عشوائى للملف غير تتابعى به ٢٠٠٠٠ سجل مخزون . افترض كذلك أن حوالى نصف السجلات فقط يتم تشغيله فى المتوسط ، فى كل مرة من مرات تنفيذ البرنامج . اذا كان الملف غير مجمع .. فإن ذلك يتطلب حوالى ١٠٠٠٠ عملية مدخلات، للحصول على ١٠٠٠٠ سجل منطقي، يجرى عليها التشغيل .

إذا كان معامل التجميع للملف هو ١٠ ، فما زال الملف فى حاجة إلى حوالى ١٠٠٠٠ عملية مدخلات للحصول على ١٠٠٠٠ سجل منطقي (لأنه لا نتوقع فى التشغيل العشوائى أن يكون السجل التالى المطلوب للتشغيل موجودا فى نفس المجموعة التى سبقت قراءتها) يمكن أن يكون وقت نقل البيانات للسجلات الإضافية (غير المشغلة) فى المجموعات مفقودا . وبالرغم من أن التجميع يمكن أن يسرع من تشغيل الملفات التتابعية .. إلا إنه يبطئ قليلا من التشغيل العشوائى كما يمكن أن يستخدم التجميع ملفات التشغيل العشوائى بفرض توفير أماكن التخزين .

يحدد BLOCK CONTAINS ما إذا كان الملف مجمعا أم لا ، ويحدد معامل التجميع . ويجمع نظام التشغيل تلقائيا السجلات المنطقية مجموعة من المخرجات، ويحلل المجموعة إلى السجلات المنطقية فى المدخلات . وعلى هذا .. يكتب جزء الاجراءات دائما للعمل مع سجلات منطقية فردية ، بغض النظر عن معامل التجميع (انظر القسم الثامن من هذا الفصل) . وتوجد ثلاثة إمكانيات :

• **الملف غير مجمع :** فى هذه الحالة .. يحذف BLOCK CONTAINS من FD .

```
FD WEEKLY-PERSONNEL-REPORT
RECORD CONTAINS 132 CHARACTERS
LABEL RECORDS ARE OMITTED
.
01 PERSONNEL-REPORT-RECORD ...
```

حيث يسرى التجميع على الأساط الممنطة فقط ؛ وإذا لا تكون ملفات البطاقات أو الطباعة مجمعة .

• **مجموعات ثابتة الطول :** يستخدم معها BLOCK CONTAINS integer-2 CHARACTERS

```
FD MASTER-INVENTORY-FILE
BLOCK CONTAINS 500 CHARACTERS
RECORD CONTAINS 100 CHARACTERS
LABEL RECORDS ARE STANDARD
```

BLOCK CONTAINS integer-2 RECORDS : أو يستخدم معها

FD MASTER-INVENTORY-FILE
BLOCK CONTAINS 5 RECORDS
RECORD CONTAINS 100 CHARACTERS
LABEL RECORDS ARE STANDARD

في المثال الاول معامل التجميع المشمول هو :

$$\frac{500 \text{ بايت / مجموعة}}{100 \text{ بايت / سجل}} = 5 \text{ سجلات / مجموعة}$$

بينما في المثال الثاني :

$$(5 \text{ سجل / مجموعة}) (100 \text{ بايت / سجل}) = (500 \text{ بايت / مجموعة})$$

والصيفتان متكافئتان تماما .

• مجموعات متغيرة الطول : وتحدث عندما تختلف اطوال السجلات المنطقية الموجودة في الملف، ويستخدم معها

BLOCK CONTAINS integer-1 TO integer-2 CHARACTERS
RECORD CONTAINS integer-3 TO integer-4 CHARACTERS

(انظر مثال ٥ - ٦) أو يستخدم معها

BLOCK CONTAINS integer-2 RECORDS
RECORD CONTAINS integer-3 TO integer-4 CHARACTERS

(انظر مثال ٥ - ٧)

مثال ٥ - ٦ :

FD MASTER-CUSTOMER-FILE
BLOCK CONTAINS 220 TO 420 CHARACTERS
RECORD CONTAINS 50 TO 100 CHARACTERS
LABEL RECORDS ARE STANDARD

يحدد شكل ٥ - ٣ تكوين إحدى المجموعات متغيرة الطول (بافتراض أن معامل التجميع هو ٣) وذلك باستخدام كود IBM OS/VS.

...	gap	BDW	RDW	Logical Record	RDW	Logical Record	RDW	Logical Record	gap	...
-----	-----	-----	-----	-------------------	-----	-------------------	-----	-------------------	-----	-----

BDW = كلمة واصف المجموعة block descriptor word (يتم إدخالها تلقائياً عن طريق المترجم) ، وتحدد طول المجموعة بما في ذلك ٤ بايت خاصة بها .

RDW = كلمة واصف سجل record descriptor word (يتم إدخالها تلقائياً عن طريق المترجم) وتحدد طول السجل المنطقي التي تسبقه بما في ذلك ٤ بايت خاصة بها .

شكل (٥ - ٣)

لا يشمل جزء RECORD CONTAINS كلمة RDW إلا أن BLOCK CONTAINS... CHARACTERS يشمل RDE, BDW وعلى ذلك... فاقصر مجموعة لهذا الملف تحتوى على :

$$\text{٤ سجلات} = \frac{٤ - ٢٢٠}{٤ + ٥٠}$$

وتحتوى أطول مجموعة للملف على :

$$\text{٤ سجلات} = \frac{٤ - ٤٢٠}{٤ + ١٠٠}$$

وإيجازاً فمعامل التجميع للملف هو ٤ .

مثال ٥ - ٧ :

FD MASTER-CUSTOMER-FILE
BLOCK CONTAINS 4 RECORDS
RECORD CONTAINS 50 TO 100 CHARACTERS
LABEL RECORDS ARE STANDARD

وهنا باعتبار مثال ٥ - ٦ فإن معامل التجميع يكون واضحا وهو ٤ . ويكون أقل طول مجموعة واقعى على ذلك هو:

$$٤ (٤ + ١٠٠) = ٤٢٠ \text{ بايت}$$

$$٤ (٤ + ٥٠) = ٢٢٠ \text{ بايت}$$

ومن الملاحظ هنا ان FD مكافئ تماما لنظيره فى مثال ٥ - ٦ .

٥ - ٤ وصف الملف وما يحتويه السجل

سبق تغطية تطبيق هذا الجزء على السجلات ثابتة الطول، والسجلات متغيرة الطول فى القسم السابق من هذا الفصل . وهذا الجزء اختياري، لان المترجم يستطيع أن يحدد طول السجل من المستوى 01 الخاص بوصف السجل (الذى يشمل حجم كل حقل) . إلا إنه من المستحسن استخدام RECORD CONTAINS دائما؛ حيث إنه يمثل توثيقا مفيدا للبرنامج كما أن بعض المترجمات تنتج رسالة تحذير ، إذا لم يتفق حجم الحقول مع قيمة هذا الجزء .

أختواء المجموعة أو السجل على صفر من الرموز

من الممكن فى كويل IBM OS/VS كتابة :

BLOCK CONTAINS 0 [CHARACTERS
RECORDS]

أو كتابة :

RECORD CONTAINS 0 CHARACTERS

وهذا هو اتساع extension لكويل ANS النمطى ، الذى يتطلب عددا موجبا من الرموز أو السجلات . ويحدد استخدام الصفر فى كويل IBM حجم المجموعة أو السجل ، الذى يجب أن يؤخذ من عبارات لغة العمل التى تصف الملف أكثر (انظر القسم الخامس من الفصل الرابع) . وبالرغم من أن RECORD CONTAINS 0 CHARACTERS له استخدامات عملية بسيطة فى الكويل إلا أن تطبيقات BLOCK CONTAINS 0 يمكن ، بل يجب ، ان تستخدم فى جعل برنامج الكويل مستقلا عن معامل التجميع (بحيث إن تغيير معامل التجميع لا يحتاج إلى تغيير أو ترجمة برنامج مصدر الكويل) . (تذكر أن ملفات البطاقات والطباعة لا يمكن أن يحدث فيها تجميع) .

مثال ٥ - ٨ :

FD OPEN-INVOICE-FILE
BLOCK CONTAINS 0 CHARACTERS
RECORD CONTAINS 250 CHARACTERS
LABEL RECORDS ARE STANDARD

يؤخذ طول المجموعة من عبارة تحكم العمل المصاحبة للملف أثناء التنفيذ . فإذا كان الملف موجودا فعلا وله عنوان ملف .. فإنه يمكن أخذ طول المجموعة كذلك من عنوان الملف (مع حذف الحاجة إلى تحديده في عبارة لغة تحكم العمل) .

٥ - ٥ وصف الملف وسجلات البيانات

تذكر من القسم رقم العاشر في الفصل الأول أن لكل ملف موجود على وحدة اتصال مباشر - عنوان ملف file label (يقدمه نظام التشغيل تلقائياً) والذي يحدد الاسم الذي يعرف به الملف لدى نظام التشغيل ، ونوع التنظيم (تتابعى أو مفهرس أو نسبى) ، وموقع منطقة الفهرس (إذا كان هناك فهرس) ، وموقع سجلات البيانات، وحجم المجموعة ، ونوع السجل (ثابت الطول أو متغير الطول) وطول السجل ، وأى كلمات مرور لازمة للاتصال بالملف ، وطول الفترة التي يجب أن يحمى فيها الملف من حذفه .. إلخ . وتجمع عناوين الملفات الموجودة على مجموعة أقراص مع بعضها البعض في ملف خاص بنظام التشغيل يسمى بحجم جدول المحتويات (VTOC) volume table of contents أو دليل directory .

إن عناوين الملفات اختيارية بالنسبة للمفات الشرائط المغناطيسية ، ولكن بسبب مميزاتها العديدة .. فهي تستخدم دائماً . وعناوين الملفات غير ممكنة لوحداث مدخلات ومخرجات الوحدة (قارئات البطاقات والمثقيات والطابعات) . الجزء LABEL RECORDS هو الجزء الوحيد المطلوب تواجده اجبارياً في FD . استخدم :

LABEL RECORDS ARE OMITTED

إذا كان الملف على وحدة منخلات ، أو وحدة مخرجات ، أو إذا ما كان الملف على شريط مغناطيسى، وليس له عناوين ملف .
وإستخدم :

LABEL RECORDS ARE STANDARD

إذا كان الملف في وحدة اتصال مباشر (قرص ، أو أسطوانة، أو قرص مرن .. إلخ) أو إذا كان الملف على شريط مغناطيسى، وله عناوين ملف .

٥ - ٦ وصف الملف وسجلات البيانات

يسرد جزء DATA RECORDS الأسماء التي يعرفها البرمج ، والتي يشار بها إلى السجلات في جزء الإجراءات . ولهذا الجزء فائدة محددة : خاصة بالنسبة للسجلات ثابتة الطول ، حيث تتبع أسماء الكويل FD مباشرة (فهى أول محتوى على المستوى 01 لوصف السجل) . ويجب أن تتفق أسماء السجلات على المستوى 01 مع الأسماء الموجودة في جزء DATA RECORDS . وجزء DATA RECORDS اختياري دائماً .

مثال ٥ - ٩ :

FD WEEKLY-TIME-FILE
BLOCK CONTAINS 436 TO 1636 CHARACTERS
RECORD CONTAINS 50 TO 200 CHARACTERS
LABEL RECORDS ARE STANDARD
DATA RECORDS ARE CLERICAL-TIME-RECORD
LAWYER-TIME-RECORD

01 CLERICAL- TIME- RECORD •

(وصف سجل منطقي طوله ٥٠ بايت)

01 LAWYER- TIME- RECORD •

(وصف سجل منطقي طوله ٢٠٠ بايت)

بافتراض كويل IBM OS/VS ، فإن معامل التجميع لـ WEEKLY- TIME- FILE هو ٨٠ .

عادة ما يكون الملفات البطاقات (غير مجمعة بالضرورة) أكثر من نوع سجل واحد نظرا لان كل الحقول المطلوبة قد لاكتفيها بطاقة واحدة ، وعندما يحدث ذلك .. فتقسم المعلومات عادة على عدة بطاقات وتكتب كل بطاقة من بطاقات المجموعة بشفرة لتحديد الحقول الموجودة فيها . وتوضع شفرة البطاقة card code عادة في أول او في آخر بايت من السجل (البطاقة)

مثال ٥ - ١٠ :

من المرغوب فيه وضع معلومات عناوين بريدية في بطاقات نمطية بها ٨٠ عموداً . ونظرا لان البيانات تتطلب أكثر من ٨٠ رمزا .. فإن الملف يحتوى على ثلاثة أنواع من السجلات . يمكن أن يكون وصف الملف FD ووصف السجل (المستوى 01) لمثل هذا الملف على النحو التالي :

FD MAILING-LABEL-FILE
RECORD CONTAINS 80 CHARACTERS
LABEL RECORDS ARE OMITTED
DATA RECORDS ARE RECORD-TYPE-1-NAME
RECORD-TYPE-2-CITY
RECORD-TYPE-3-BUSINESS

01	RECORD-TYPE-1-NAME.	
05	CODE-1	PIC X.
05	ID-1	PIC X(9).
05	NAME-1	PIC X(30).
05	STREET-1	PIC X(30).
05	FILLER	PIC X(10).
01	RECORD-TYPE-2-CITY.	
05	CODE-2	PIC X.
05	ID-2	PIC X(9).
05	PHONE-2	PIC X(8).
05	CITY-2	PIC X(30).
05	STATE-2	PIC X(2).
05	ZIP-2	PIC X(9).
05	FILLER	PIC X(21).
01	RECORD-TYPE-3-BUSINESS.	
05	CODE-3	PIC X.
05	ID-3	PIC X(9).
05	BUSINESS-3	PIC X(70).

لاحظ ان جزء BLOCK CONTAINS محذوف الملف البطاقات غير المجمع . وقد استخدم جزء RECORD CONTAINS حتى يصدر الكمبيوتر رسالة تحذيرية إذا لم يكن مجموع خانات أى سجل على المستوى 01 مساوياً ٨٠ (وهذا اختبار مفيد ، حيث انه لا يوجد اتجاه لوجود سجلات متغيرة الطول ، ولكن توجد ثلاثة أنواع من السجلات طول كل منها ثابت وهو ٨٠ خانة) . يحذف جزء LABEL RECORDS الملف البطاقات . يعمل جزء DATA RECORD كتنسيق للبرنامج ؛ ليسهل تحديد عدد الأنواع المختلفة من السجلات الموجودة فى الملف ، هل تستطيع تسمية الحقول فى كل نوع من أنواع السجلات المنطقية ؟

٥ - ٧ وصف الملف والخطية

يستخدم جزء الخطية LINAGE للمفات المخرجات المطبوعة فقط وهو اختياري دائما . وعند استخدامه.. فإنه يعرف صفحة منطقية logical page ، تشمل هامشاً علوياً top margin ، وهامشاً سفلياً bottom margin وجسم للصفحة page body تكتب داخله الطباعة الفعلية . ويمكن أن تعرف منطقة فى أسفل جسم الصفحة بطريقة اختيارية، بأنها منطقة النهاية footing area .

يمكن استخدام الخطية LINAGE بدلاً من قنوات تحكم العربية (القسم الرابع من الفصل الرابع) ولا يمكن استخدام الطريقتين لنفس الملف . ولايتاح جزء LINAGE مع كل المترجمات . انظر المشكلة رقم ٢٠ من الفصل الثامن للتعرف على أساليب التحكم فى الصفحات بدون LINAGE .

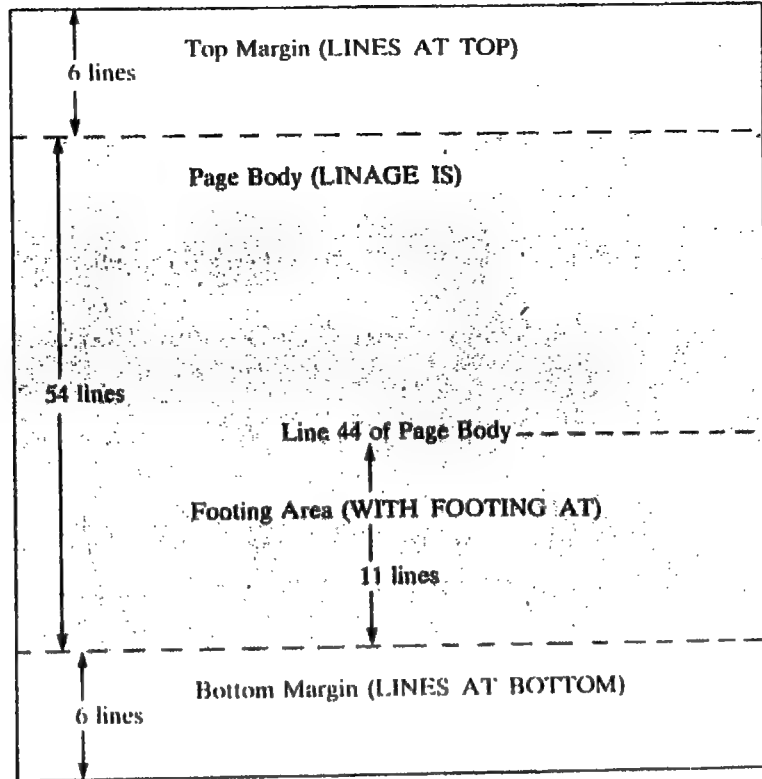
مثال ٥ - ١١ :

```
FD SALES-BY-SALESPERSON-REPORT
RECORD CONTAINS 132 CHARACTERS
LABEL RECORDS ARE OMITTED
LINAGE IS 54 LINES
WITH FOOTING AT 44
LINES AT TOP 6
LINES AT BOTTOM 6
```

ملف الطباعة غير مجمع ، وليس له عنوان ملف . يعرف جزء الخطية LINAGE صفحة منطقية ، كما هو محدد في شكل ٥ - ٤ .

وتحدث كل الطباعة داخل جسم الصفحة (الجزء المظلل) .
دائماً يترك الهامشان العلوى والسفلى من الصفحة فارغين دائماً ، وعلى هذا تخدم الهوامش في فصل جسم الصفحة من جسم الصفحة التالى والسابق له . الجزءان : LINES AT TOP و LINES AT BOTTOM اختياريان إذا ما حذف أى منهما فيفترض في الهامش المناظر له أنه لا يحتوى على أى أسطر .
يمكن استخدام منطقة النهاية footing area داخل جسم الصفحة في :

- (١) طباعة إحصائيات الصفحة لحقول معينة .
- (٢) طباعة رسالة عندما يتقاطع جزء من التقرير مع حدود الصفحة، ويجب على هذا أن يستمر في جسم الصفحة التالى.
- (٣) التأكد من وجود مكان كاف في نهاية جسم الصفحة ، لأنهاء أى مادة تكون قد بدأت (وعلى هذا حذف إمكانية استمرار شيء معين على الصفحات التالية) . منطقة النهاية اختيارية وإذا ما حذف الجزء WITH FOOTING AT فيفترض أن منطقة النهاية تحتوى على صفر من الاسطر في نهاية جسم الصفحة .



شكل (٥ - ٤)

تطبع في جسم الصفحة أسطر التقرير الطبيعية . انظر الفصل السادس لمناقشة جزء الإجراءات عند استخدام عبارة WRITE في إخراج ملف مع استخدام LINAGE .

مثال ٥ - ١٢ :

- FD OUT-OF-STOCK-REPORT
RECORD CONTAINS 132 CHARACTERS
LABEL RECORDS ARE OMITTED
LINAGE IS 66 LINES

يبلغ طول كل جسم صفحة ٦٦ سطرا ويجاور جسم الصفحة التالي مباشرة . منطقة النهاية بها صفر من الاسطر (أى إنها غير موجودة فى الصفحة) إلا أن اكتشاف منطقة نهاية سوف يحدد بالإشارة إلى عبارة WRITE عند الوصول إلى نهاية جسم الصفحة .

- FD SALES-BY-CUSTOMER-REPORT
RECORD CONTAINS 132 CHARACTERS
LABEL RECORDS OMITTED
LINAGE 50 LINES
LINES AT TOP SIZE-TOP-MARGIN
LINES AT BOTTOM 5

أى من ، أو كل ، أجزاء LINAGE يمكن أن يحدد قيمة توضع فى تخزين قبل فتح OPEN الملف (قبل إعداده للتشغيل ، انظر مثال مثال رقم ٥ من الفصل الثانى) . فى هذه الحالة ، الرقم الذى ليس له إشارة الموجود فى موقع SIZE- TOP- MARGIN عند فتح OPEN الملف SALES- BY- CUSTOMER- REPORT يكون عبارة عن عرض الهامش العلوى . تسمح هذه السمة للبرنامج بإدخال أو حساب عرض أى من مناطق الخطية LINAGE .

٥ - ٨ وصف السجل فى قسم الملفات

عندما يقوم برنامج هدف كويل بتشغيل ملف .. فيجب أن تحجز منطقة من ذاكرة الكمبيوتر لاستقبال السجل الواقعى (من ملف المدخلات) ، أو لبناء السجل الواقعى (الملف المخرجات) ، وتسمى مثل هذه المنطقة بالذاكرة الاحتياطية buffer الملف معين . والغرض من جزء وصف السجل هو وصف السجل المنطقى النشط حالياً ، كما يظهر فى الذاكرة الاحتياطية . وفى كلمات أخرى ، يحدد جزء وصف السجل هذا الجزء من الذاكرة الاحتياطية الذى يمسك بالسجل المنطقى ، الذى يجرى عليه التشغيل حالياً .

مثال ٥ - ١٢ :

لجزء برنامج مثال ٥ - ٩ الاستمرارية التالية :

- FD WEEKLY-TIME-FILE
BLOCK CONTAINS 436 TO 1636 CHARACTERS
RECORD CONTAINS 50 TO 200 CHARACTERS
LABEL RECORDS ARE STANDARD
DATA RECORDS ARE CLERICAL-TIME-RECORD
LAWYER-TIME-RECORD

01 CLERICAL- TIME- RECORD •

(وصف سجل منطقي طوله ٥٠ بايت)

01 LAWYER- TIME- RECORD •

(وصف سجل منطقي طوله ٢٠٠ بايت)

....
PROCEDURE DIVISION.
....
OPEN INPUT WEEKLY-TIME-FILE
....
READ WEEKLY-TIME-FILE RECORD
....

توضح عبارات جزء الإجراءات كيفية إعداد ملف للمسجلات (OPEN INPUT WEEKLY- TIME- FILE) ، وكيفية الإدخال الفعلي لسجل من الملف READ WEEKLY- TIME- FILE RECORD . وتحقق عبارة READ مايلي:

(١) إذا كانت الذاكرة الاحتياطية فارغة ، فإنها تدخل سجلا واقعيًا ، وتجعل أول سجل منطقي - في هذا السجل الواقعي - متاحًا للبرنامج .

(٢) إذا كانت الذاكرة الاحتياطية تمسك حاليًا بسجل واقعي ، وتم إجراء تشغيل على كل سجلاته المنطقية .. فإنها تجرى نفس الإجراء مثل الحالة الأولى .

(٣) إذا كانت الذاكرة الاحتياطية تمسك سجلا واقعيًا ، لم يتم تشغيل كل سجلاته المنطقية ، فإنها تجعل السجل المنطقي التالي (الذي لم يجر عليه تشغيل) متاحًا للبرنامج .

في أي حالة من الحالات ، بعد عبارة READ ، يكون هناك سجلا منطقيًا متاحًا للبرنامج . ويرجع الأمر إلى المبرمج ، ليتأكد ما إذا كان البرنامج يميز أي نوع من السجلات المنطقية . والإجراء العادي - هنا - هو حجز بايت معين في كل نوع من أنواع السجلات كشفرة للسجل record code . وعلى هذا .. فأول بايت لكل سجلات الكتبة clerical يمكن أن يحتوى على الحرف C ، بينما يمكن أن يحتوى أول بايت ، من كل السجلات للمحامين lawyer على الحرف L . أول السجل الذي له حرف C ، يجرى عليه تشغيل C بواسطة البرنامج ، كما أن البرنامج يجرى تشغيل L على السجل الذي له الحرف L .

أرقام المستويات

كجزء أساسي من وصف السجل .. تحدد أرقام مستويات للحقول الموجودة في السجل المنطقي ، وتتراوح من 01 الى 49 والمستوى 01 محجوز لاسم السجل نفسه . إذا ازداد رقم المستوى عند الانتقال من حقل لآخر .. فيقال أن الحقل الثاني هو حقل جزئي subfield من الحقل الأول .

مثال ٥ - ١٤ :

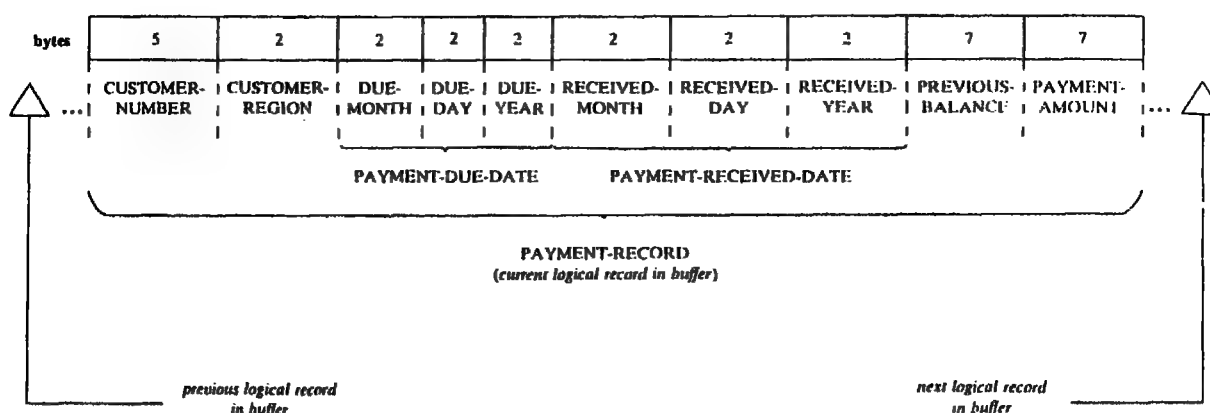
FD PAYMENT-FILE
BLOCK CONTAINS 100 RECORDS
RECORD CONTAINS 33 CHARACTERS
LABEL RECORDS ARE STANDARD

01 PAYMENT-RECORD.
05 CUSTOMER-NUMBER PIC X(5).
05 CUSTOMER-REGION PIC X(2).
05 PAYMENT-DUE-DATE.

10	DUE-MONTH	PIC X(2).
10	DUE-DAY	PIC X(2).
10	DUE-YEAR	PIC X(2).
05	PAYMENT-RECEIVED-DATE.	
10	RECEIVED-MONTH	PIC X(2).
10	RECEIVED-DAY	PIC X(2).
10	RECEIVED-YEAR	PIC X(2).
05	PREVIOUS-BALANCE	PIC S9(5)V99.
05	PAYMENT-AMOUNT	PIC S9(5)V99.

يحدد رقم المستوى 01 المكتوب في المنطقة A ، بداية جزء وصف السجل . وتكتب أرقام المستويات الاخرى ، وهي 05 ، 10 هنا ، في المنطقة B . لاحظ استخدام الترحيل في جزء البيانات وذلك بالرغم من أن الترحيل غير مطلوب بالنسبة للمترجم ، إلا إنه يمثل نمطية كتابة شفرة مهمة جدا والتي توضح تكوين السجل .

الهيكل المعرف بواسطة أرقام مستويات المثال الحالي مرسوم في شكل ٥ - ٥ .



شكل (٥ - ٥)

تسمى الحقول التي تشتمل على حقول جزئية مجموعة عناصر group items ، والحقول التي لا تشتمل على حقول جزئية تسمى عناصر فردية elementary items .

group items	PAYMENT-RECORD	مثال ٥ - ١٥ :
	PAYMENT-DUE-DATE	
	PAYMENT-RECEIVED-DATE	
elementary items	CUSTOMER-NUMBER	
	CUSTOMER-REGION	
	DUE-MONTH	
	DUE-DAY	
	DUE-YEAR	
	RECEIVED-MONTH	
	RECEIVED-DAY	
	RECEIVED-YEAR	

لاحظ أن العناصر الفردية لها جزء PIC يصف طول ونوع البيانات للعنصر . وليس لمجموعة العناصر مثل هذا الوصف؛ لأنها معرفة تماما بواسطة العناصر الفردية التي تتكون منها . وتشير الاسماء التي يعرفها المبرمج دائما إلى مجموعات العناصر والعناصر الفردية إلى سجل منطقي نشط حاليا في الذاكرة الاحتياطية.

وكتابة الشفرة النمطية التي يوصى بها، هي استخدام اختصار لاسم السجل (الاسم الموجود على مستوى 01) أو اختصار لاسم الملف (الاسم الموجود على مستوى FD) كسابقة لكل اسم بيانات موجود في السجل . على هذا يصبح من الواضح إلى أي ملف أو سجل ينتمي عنصر بيانات معين .

مثال ٥ - ١٦ :

```
FD MAILING-LABEL-FILE
RECORD CONTAINS 76
LABEL RECORDS OMITTED
DATA RECORDS MAILING-LABEL-NAME-STREET-REC
MAILING-LABEL-CITY-STATE-REC
```

01 MAILING-LABEL-NAME-STREET-REC.

05 MAILING-LABEL-NAME-ST-CODE	PIC X.
05 MAILING-LABEL-NAME-ST-ID	PIC X(5).
05 MAILING-LABEL-NAME-ST-NAME	PIC X(30).
05 MAILING-LABEL-NAME-ST-STREET	PIC X(30).
05 FILLER	PIC X(10).

01 MAILING-LABEL-CITY-STATE-REC.

05 MAILING-LABEL-CITY-STATE-CODE	PIC X.
05 MAILING-LABEL-CITY-STATE-ID	PIC X(5).
05 MAILING-LABEL-CITY-STATE-PHONE	PIC X(8).
05 MAILING-LABEL-CITY-STATE-CITY	PIC X(30).
05 MAILING-LABEL-CITY-STATE-STATE	PIC X(2).
05 MAILING-LABEL-CITY-STATE-ZIP	PIC X(9).
05 FILLER	PIC X(21).

هناك مستويان من السابقات، يستخدمان فعلا في هذا المثال ، وهما:

(١) كل اسم سجل، واسم حقل يبدأ بـ MAILING- LABEL- FILE ، وهو اختصار لاسم الملف MAILING- LABEL- FILE . أى اسم بيانات يبدأ بهذه السابقة .. يكون معروفا على هذا بأنه مصاحب للملف MAILING- LABEL- FILE . وهذا .. يجعل من الممكن قراءة جزء الإجراءات دون الإشارة المستمرة إلى جزء البيانات لمعرفة أى سجل أو ملف يحتوى على حقل معين . (عندما يكون هناك أكثر من مستوى واحد 01 لنفس مستوى FD .. يقال عن أوصاف السجلات المنطقية إنها تعيد تعريف نفسها ضمنيا).

(٢) يحتوى اسم كل حقل في السجل MAILING- LABEL- NAME- STREET- REC على السابقة الثانية NAME- STREET- REC ، وهى اختصار NAME- STREET- REC (مثال ذلك MAILING- LABEL- NAME- ST- CODE يصاحب MAILING- LABEL- FILE . ويصفه خاصة .. هناك حقلان من السجل MAILING- LABEL- NAME- STREET- REC في هذا الملف) . وبالمثل .. لكل حقل في السجل MAILING- LABEL- CITY- STATE- REC سابقة ثانية ، وهى CITY- STATE في اسمه.

مثال ٥ - ١٧ :

يوجد واصفان لمستويات 01 في مثال ٥ - ١٦ للسجل المنطقي النشط حاليا في الذاكرة الاحتياطية ، وهذان الوصفان موضحان في شكل ٥ - ٦ .

لاحظ أن كلا من واصفى السجل متفقان مع تفسير أول ٦ بايت من السجل المنطقي . وكما يشير MAILING- LABEL- NAME- ST- CODE و MAILING- LABEL- CITY- STATE- CODE إلى نفس موقع الذاكرة (بالاسم) أول بايت من السجل المنطقي) ، فيمكن أن يعالج اسما البيانات هذين كمرافقين خلال بقية البرنامج . وبالمثل .. يشير MAILING- LABEL- NAME- ST- ID و MAILING- LABEL- CITY- STATE- ID إلى البايت من ٢ - ٦ في السجل المنطقي .. وعلى هذا .. فتبادل استخدام اسمى البيانات في جزء الإجراءات لا يؤثر على تنفيذ البرنامج . يمكن استخدام أول بايت من السجل المنطقي (والتي يمكن الإشارة إليها بإنها MAILING- LABEL- NAME- ST- CODE أو بإنها MAILING- LABEL- CITY- STATE- CODE لتحتوى على شفرة سجل تعرفه كسجل اسم وشارع أو كسجل مدينة وولاية .) انظر مثال رقم ١٣ والمشكلة رقم ٨٠ من هذا الفصل).

	NAME- ST- CODE	NAME- ST- ID	NAME- ST- NAME		NAME- ST- STREET		unused	
bytes	1	5	30		30		10	
...	currently active logical record in buffer							
bytes	1	5	8	30		2	9	21
	CITY- STATE- CODE	CITY- STATE- ID	CITY- STATE- PHONE	CITY- STATE- CITY		CITY- STATE- STATE	CITY- STATE- ZIP	unused

شكل (٥ - ٦)

الذاكرات الاحتياطية المتعددة

بالنسبة للملفات الاتصال التابعى .. يمكن أن يكون هناك ميزة وجود ذاكرتين احتياطيتين (أو أكثر) : يجرى تشغيل على محتويات ، احداها داخل وحدة التشغيل المركزية، مع تغذية الثانية بالمجموعة الثانية من وحدة المدخلات والمخرجات فى نفس وقت التشغيل . وعدد الذاكرات الاحتياطية التقليدى للملفات الاتصال العشوائى فهو واحد . ويوجد جزء إضافى فى كويل ANS فى عبارة SELECT ، يسمح للمبرمج بتحديد عدد الذاكرات الاحتياطية الملف كرقم صحيح .

SELECT ... ASSIGN TO ... [RESERVE integer [AREA]] ...

ربما يكون من الأفضل - إلا إذا كنت معتادا جدا على نظام الكمبيوتر، وتكون المدخلات والمخرجات - حذف جزء RE-SERVE ، وقبول العدد التقليدى للذاكرات الاحتياطية الذى يقدمه المترجم .

٩ - ٥ تكوين وصف البيانات

الغرض من محتوى وصف البيانات فى قسم الملفات هو الوصف التفصيلى لحقل السجل . يحدد شكل ٥ - ٧ التكوين العام لمحتوى وصف البيانات

- level-number { data-name-1 }
 FILLER
- (1) [REDEFINES data-name-2]
 - (2) [{ PICTURE } IS character-string]
 PIC
 - (3) [[USAGE IS] { DISPLAY
 COMPUTATIONAL
 COMP
 COMPUTATIONAL-3
 COMP-3 }]
 - (4) [BLANK WHEN ZERO]
 - (5) [{ JUSTIFIED } RIGHT]
 JUST
 - (6) [OCCURS integer TIMES]
 - (7) [{ SIGN IS } { LEADING } SEPARATE CHARACTER]
 TRAILING
 - (8) [{ SYNCHRONIZED } { LEFT }]
 SYNC RIGHT
 - (9)

شكل (٥ - ٧)

يجب أن يقع رقم المستوى بين 01 و 49 ، حيث يحجز الرقم 01 لأسماء السجلات . Data- name-1 هو أسم يعرضه المبرمج للحقل ، لاستخدامه في عبارات جزء الإجراءات التي تجرى تشغيلاً على الحقل . فإذا كان موجوداً أحد الحقول ولن يجرى البرنامج تشغيلاً عليه ، فيمكن استخدام كلمة FILLER من مكان اسم الحقل . ويمكن استخدام FILLER كلما كانت هناك حاجة لذلك .

مثال ٥ - ١٨ :

```
FD PAYMENT-FILE
  LABEL RECORDS ARE STANDARD.
01 PAYMENT-RECORD.
  05 CUSTOMER-NAME          PIC X(20).
  05 PAYMENT-AMOUNT          PIC S9(5)V99.
FD WEEKLY-RECEIPTS-REPORT
  LABEL RECORDS ARE OMITTED.
01 WEEKLY-RECEIPTS-RECORD.
  05 CUSTOMER-NAME          PIC X(20).
  05 FILLER                  PIC X(5).
  05 PAYMENT-AMOUNT          PIC ZZ.ZZZ.99-.
  05 FILLER                  PIC X(65).
.....
PROCEDURE DIVISION.
.....
  READ PAYMENT-FILE RECORD
  MOVE CUSTOMER-NAME OF PAYMENT-RECORD
    TO CUSTOMER-NAME OF WEEKLY-RECEIPTS-RECORD
  MOVE PAYMENT-AMOUNT OF PAYMENT-RECORD
    TO PAYMENT-AMOUNT OF WEEKLY-RECEIPTS-RECORD
```

تسمح قواعد لغة الكوبل بازدواج أسماء البيانات : فقد ظهر كل من CUSTOMER- NAME و PAYMENT- AMOUNT في سجلين مختلفين في هذا المثال . عندما يتكرر استخدام اسم بيانات في جزء الإجراءات فيجب أن يؤهل qual- ified أى يجب أن يكون تابعا لمجموعة عناصر فريدة، مثل :
PAYMENT- AMOUNT OF PAYMENT CUSTOMER- NAME OF WEEKLY- RECEIPTS- RECORD,
ونوصى باستخدام أسماء بيانات فردية مع استخدام سابقات من أسماء الملفات أو السجلات (مثال ٥ - ١٦) بدلا من استخدام تكرار الأسماء وتأهيل هذه الأسماء .

إذا استخدم جزء REDEFINES فيجب أن يتبع رقم المستوى مباشرة ، أما بقية الأجزاء فيمكن أن تظهر بأي ترتيب .
ونناقش في الأقسام التالية من ١٠ إلى ١٧ هذه الأجزاء بترتيب الأكثر استخداما أولا، فالأقل استخداما . (باستثناء موقع RE-
DEFINES فهذا له الترتيب الموضح في شكل ٥ - ٧ .

٥ - ١٠ جزء الصورة

يصف هذا الجزء شكل العنصر الفردي ، ولا يمكن تحديده لمجموعة عناصر . ويمكن أن تشمل سلسلة الرموز أيًا من رموز الصورة picture characters التالية :

A B P S V X Z 0 9 / , . + - CR DB * \$

تحدد سلسلة الرموز (١) عدد مواقع الرموز في العنصر الفردي (٢) ونوع البيانات (عددي أو حرفي عددي أو حرفي) الذي يشغل مواقع الرموز (٣) ، وما إذا كان عنصر البيانات يجري عليه تنقيح يبدو في صورة مرئية جيدة عند طباعته أو عرضه على الشاشة أم لا .

وجزء PICTURE معقد لوجود عديد من البدائل . ونناقش كل رمز فردي لعموده فيما يلي . تذكر أنه يجب أن يكون لكل عنصر فردي جزء صورة PICTURE .

الرمز A

يمثل كل رمز A في سلسلة الرموز موقعًا (بايت) حرفيًا، يمكن أن يحتوي على حرف أبجدي أو فراغ فقط .

مثال ٥ - ١٩ :

```
01 EMPLOYEE-NAME.
05 EMPLOYEE-LAST-NAME PIC A(15).
05 EMPLOYEE-FIRST-NAME PIC A(10).
05 EMPLOYEE-MIDDLE-INITIAL PIC A.
```

تعليقات : (١) لا توجد صورة لمجموعة العناصر EMPLOYEE- NAME .

(٢) لكل عنصر فردي جزء PICTURE وصفي .

(٣) يبسط استخدام معامل التكرار من كتابة PICTURE :

(10) A في اختصار لكتابة A عشر مرات .

(4) A في اختصار لكتابة A أربعة مرات .

(٤) يشغل EMPLOYEE- LAST- NAME عدد ١٥ موقعًا من مواقع الرموز (بايت) ، ويشغل EMPLOY-

EE- FIRST- NAME خمسة مواقع للرموز بينما يشغل EMPLOYEE- MIDDLE- INITIAL موقعًا

واحدًا فقط .

- (٥) سوف تحتوي عناصر البيانات الثلاث كلها على حروف هجائية أو فراغات فقط .
- (٦) استخدام PIC بدلا من DICTURE أمراً عادياً .

الرمز B (وهو تنقيح)

يمثل كل رمز B في سلسلة PICTURE بايت واحداً يدخل فيه فراغ عندما تنتقل البيانات إلى الحقل . والرمز B هو مثال لرمز تنقيح editing character ، يتسبب في إعادة ترتيب البيانات أو إعادة تشكيلها عندما تنتقل في حقل صورة تنقيح .
مثال ه - ٢٠ :

- 01 GREETING PIC AABA(5).

تعليقات : (١) GREETING هو حقل مكون من ٨ بايت : ٢ منهما فراغات، أو حروف يتبعهما فراغ B يتبعه ه فراغات أو حروف (5) A .
(٢) ينتج عن MOVE "HITHERE" TO GREETING المحتويات التالية في GREETING .

"HI THERE"

تتسبب B في وجود فراغ بين أول حرفين ، وبقيّة الحروف الأخرى .

- 01 EMPLOYEE-INITIALS.
05 COMPRESSED-INITIALS PIC A(3).
05 SEPARATED-INITIALS PIC ABABAB.

تعليقات : (١) لا يمكن تكرار مجموعة من رموز الصورة إلا بإعادة كتابتها (3) AB ينتج عنها ABAB ، وليس ABABAB .
(٢) إذا ما احتوت COMPRESSED- INITIALS IF على DAF فتضع

MOVE COMPRESSED-INITIALS TO
SEPARATED-INITIALS

. SEPARATED- INITIALS في DAF

الرمز X

يمثل رمز الصورة الذي يستخدم بكثرة شديدة بايت يمكن أن يحتوي على أى رمز من مجموعة رموز الكمبيوتر (مثل نظام EBCDIC لنظم IBM-370) . يستخدم عادة X في جزء PICTURE الذي يجب أن يصاحب سجل على المستوى 01 ولا يكون مقسماً إلى حقول .

مثال ٥ - ٢١ :

05 INPUT-SOCIAL-SECURITY-NUMBER	PIC X(9).
05 PRINT-SOCIAL-SECURITY-NUMBER	PIC XXXBXXBXXXX.

يمكن أن يحتوى INPUT- SOCIAL- SECURITY- NUMBER على قيمة، يتم إدخالها من قرص مثل "111223333"، رقم ضمان اجتماعي، لعامل بدون فراغات أو تنقيط غير ضروري . وتحدد PIC X (9) مكانا كافيا لوضع أي ٩ رموز . وعندما يراد طباعة رقم الضمان الاجتماعي فإننا نريد تنقيط edit القيمة لجعلها سهلة القراءة . ويمكن عمل ذلك بإدخال فراغات عن طريق رمز الصورة B لإظهار الطباعة في الصورة التالية : «111 22 3333» .

وبصفة عامة .. عندما تستخدم بيانات عديدة في التعريف بدلا من الحسابات (مثل رقم الضمان الاجتماعي أو رقم الهاتف) فمن الأفضل استخدام رمز الصورة X .

الرمز 9

يمثل كل رمز 9 في الصورة موقعا لرقم عشري (من صفر الى ٩) . على عكس الرمز A ، والرمز B ، والرمز X ، لا يتطلب الرمز 9 بايت واحد دائما (رمزا واحدا دائما) من الذكرة . الحجم الفعلي لعنصر عددي ، والذي يعتمد على شفرة البيانات ، يعالج في القسم رقم 18 من هذا الفصل . بينما لاتصح برمجة الحسابات في الكوبل باستخدام إعداد لها صورة X ، وذلك لأن الصورة 9 مصممة خصيصا للحسابات . لاتحدث 9 PIC أي تنقيط لعناصر البيانات العددية فهي لاتدخل بصفة - خاصة - تنقيطا أو تحذف أصفارا رائدة leading zeros .

مثال ٥ - ٢٢ :

01 EMPLOYEE-INFORMATION-RECORD.	
05 EMPLOYEE-INFO-SOCIAL-SECURITY	PIC X(9).
05 EMPLOYEE-INFO-DEPENDENTS	PIC 99.
05 EMPLOYEE-INFO-SICK-DAYS-LEFT	PIC 9(3).
05 EMPLOYEE-INFO-VACATION-LEFT	PIC 9(3).
01 SAVINGS-ACCOUNT-RECORD.	
05 SAVINGS-ACCT-NUMBER-DEPOSITS	PIC 9(5).
05 SAVINGS-ACCT-NUMBER-WITHDRAWS	PIC 9(5).
05 SAVINGS-ACCT-NUMBER	PIC X(7).

01 INVENTORY-MASTER-RECORD.

05 INVENTORY-MAST-QUANTITY-ONHAND PIC 9(7).

استخدم SAVINGS- ACCT- NUMBER و EMPLOYEE- INFO- SOCIAL- SECURITY ، للتعريف ، وليس الحسابات ، وعلى هذا استخدمت صورة X . أما بقية الحقول الأخرى فيمكن أن تجرى عليها حسابات ، ولهذا أعطيت صورة 9 . لاحظ إنه إذا نقلت 52 إلى حقل QUANTITY- ONHAND فإنها تخزن على النحو التالي 0000052 .

الرمز S

يحدد رمز S للصورة وجود إشارة عملية operational sign في العدد . ولاتشغل إشارة العملية أى موقع تخزين إضافي؛ لأنها تدخل في التمثيل الداخلى للبيانات للرقم (انظر القسم الحادى عشر من الفصل الأول والقسم رقم ١١ من هذا الفصل).
لتمم الأزواج الثنائية (COMP) Binary- twos- complement ، والعشرى المضغوط (COMP-3) إشارات عمليات مبنية داخل تمثيل البيانات ، وعلى هذا بالنسبة إلى USAGE فإن حذف S من سلسلة الصورة يكافئ طلب القيمة المطلقة absolute value . ويحجز تمثيل DISPLAY (من نظام EBCDIC ونظام ASCII) مكانا تلقائيا لإشارة العملية كجزء من تمثيل الرقم الموجود على أقصى اليمين، إلا أن استخدام هذا المكان أو عدمه يعتمد أساسا على ما إذا كانت هناك S موجودة فى PIC أم لا.

مثال ٥ - ٢٢ :

05 SECONDS-BEFORE-LAUNCH PIC S9(5).

تعليقات : (١) إذا استخدمت S .. فيجب أن تكون أول رمز فى PICTURE .
(٢) عند طباعة أعداد DISPLAY لها إشارات عمليات ، فيطبوع الرقم الموجود على أقصى اليمين كحرف أو كفاغ . وعلى هذا ففى تمثيل EBCDIC (جدول ٥ - ١) إذا طبعت SECONDS- BEFORE- LAUNCH الرقم 17 ثانية قبل الهبوط .. فإنها تطبع على النحو التالى : 00016 بينما إذا ما طبعت 12 ثانية بعد الهبوط فإنها تطبع على النحو التالى

0001K:

الرقم الموجود فى أقصى اليمين	يطبع مع إشارة + على النحو التالى	يطبع مع إشارة - على النحو التالى
()	فراغ	فراغ
1	A	J
2	B	K
3	C	L
4	D	M

N	E	5
O	F	6
P	G	7
Q	H	8
R	I	9

(٢) يجب أن تنقل الأعداد التي لها إشارات عمليات إلى حقول تنقيح وتطبع في صورتها المنقحة لتجنب صعوبة تفسير المخرجات المطبوعة .

مثال ٥ - ٢٤ :

05 TOTAL-BALANCE-DUE PIC 9(5).

بدون S في صورة العنصر العددي .. ينتج الكمبيوتر إشارة موجب دائما لهذا العدد معطيا القيمة المطلقة له . ولا يتطلب هذا تعليمات لغة آلة إضافية في كل مرة يعالج فيها العدد فقط بل يمكن كذلك أن يكون خطيرا . فمثلا بافتراض أن الحقل -TOTAL-BALANCE-DUE يحتوى على 00015 : محدد أن العميل مدينا بمبلغ 15 دولار . إذا دفع العميل 20 دولارا وطرحت القيمة من -TOTAL-BALANCE-DUE فيحدث مايلي : 00015 مطروحا منها 00020 تعطى 0005 - وحيث انه لا توجد S في الصورة فتخزن على انها 0005 + . وعلى هذا يبدو أن العميل مازال مدينا بمبلغ 5 دولارات بينما يكون دائما بهذا المبلغ! من الأفضل ، إذا لم تكن القيمة المطلقة هي المطلوبة ، تحديد الإشارة في الصورة دائما .

05 TOTAL-BALANCE-DUE PIC S9(5).

الرمز V

يستخدم رمز الصورة V في تحديد وجود علامة عشرية مفترضة assumed decimal point في عنصر عددي . ويستخدم بالاتصال مع رمز الصورة 9 ويمكن أن يظهر مرة واحدة فقط في PICTURE . إذا ماحذف الرمز V يفترض أن العلامة العشرية تتبع الرقم الموجود في أقصى اليمين من العدد . وحيث أن العلامة العشرية الفعلية لا تتواجد في التمثيل الداخلي للعنصر ، فإنها لا تستهلك موقع تخزين إضافي في الذاكرة . تستخدم العلامات العشرية المفترضة في ضبط العناصر العددية لأغراض التنقيح أو الحسابات .

مثال ٥ - ٢٥ :

05 HOURS-WORKED-THIS-WEEK PIC S99V9.
05 TOTAL-HOURS-FOR-YEAR PIC S9(4).

يحتوى حاليا على $42_{\wedge}5 + 0063_{\wedge}+$ على التوالى (يكون جزء من الساعة معنويا بالمقارنة بالاسبوع ، ولكنه لا يكون كذلك بالنسبة للسنة) . إذا كان مطلوبا من الكمبيوتر أن يضيف ساعات الاسبوع إلى ساعات السنة ، فيجب أن يذكر له موقع العلامة العشرية .

صحيح	غير صحيح
ساعات الاسبوع = $42_{\wedge}5$	ساعات الاسبوع = 425
ساعات السنة = 0603_{\wedge}	ساعات السنة = 0603
المجموع $5_{\wedge}0645$	المجموع 1028

إذا كانت النتيجة الصحيحة ستخزن في اسم البيانات TOTAL- HOURS- FOR- YEAR .. فيجب أن يحذف الجزء الكسرى (5 ر) ، أو يجب أن يقرب العدد . إذا حذف الكسر العشري باستخدام PIC S9(4) فتكون النتيجة 0645+ ، ويضيف التقريب 1 إلى الرقم الموجود على أقصى اليمين لحفظه إذا كان الرقم الموجود أقصى اليسار الذى يحذف 5 أو أكثر . فإذا قربت $5_{\wedge}0645$ إلى PIC S9(4) فتصبح النتيجة 0646+ .

الرمز Z

يستخدم رمز الصورة Z بدلا من الرمز 9 لتمثيل موقع الرقم العشري المراد استبداله بفراغ إذا ما احتوى هذا الموقع على صفر من الاصفار الرائدة . لا يمكن استخدام الحقول المعرفة برموز تنقيح (بما فى ذلك Z) فى الحسابات ، وذلك بالرغم من إنها يمكنها أن تستقبل نتائج الحسابات .

مثال ٥ - ٢٦ :

- 05 PRICE PIC ZZZZ9.

إذا احتوى PRICE على القيمة 203 وكان جزءاً من سجل مخرجات على الطابع ، فإنه يطبع على النحو التالى 203 bb (b تعنى موقعا فارغا) . أول رمزين Z فى PIC مناظرين لأصفار رائدة فى القيمة المطبوعة ، وعلى هذا يترك فراغ فى هذين الموقعين.

الرمز Z الرابع فى PIC يناظر 0 فى القيمة ، كذلك إلا إنه ليس صفرا رائدا (أى إنه لايسبق كل الأرقام غير الصفرية). وعلى هذا .. يطبع الصفر (إذا احتوى PRICE على أصفار فالقيمة المطبوعة تصبح bbbb0 حيث أن 9 تمثل صفر دائما سواء كان هذا الصفر رائدا أم لا) .

- 05 PRICE PIC ZZZZ.

إذا احتوى PRICE على صفر، وطبعت قيمته فإنها تظهر على النحو التالى : bbbb .

رمز تنقيح النقطة

تمثل النقطة (العلامة العشرية) الموقع الفعلي للعلامة العشرية في الحقل (المناظرة لموقع V ، التي تمثل علامة عشرية افتراضية في الصورة) . وتشغل النقطة " . " بايت واحداً من الحقل في الذاكرة وتظهر عند طباعة الحقل أو ظهوره على الشاشة . وتعتبر النقطة رمز تنقيح ، لذا فلا تستخدم الحقول الموجودة في صورتها نقطة في الحسابات ، بل يمنع كذلك ظهور النقطة كأخر رمز في الصورة .

مثال ٥ - ٢٧ :

01	INVOICE-DETAIL-LINE.	
05	INVOICE-PART-NUMBER	PIC X(5).
05	FILLER	PIC X(8).
05	INVOICE-PART-DESCRIPTION	PIC X(20).
05	FILLER	PIC X(5).
05	INVOICE-QUANTITY-PURCHASED	PIC ZZ9.
05	FILLER	PIC X(3).
05	INVOICE-UNIT-PRICE	PIC ZZZ.99.
05	FILLER	PIC X(10).
05	INVOICE-TOTAL-PRICE	PIC ZZZZZZ.99.

تعليقات :

- (١) مجموعة العناصر (المستوى 01 ليس لها PICTURE) .
- (٢) الحقل INVOICE- PART- NUMBER هو حقل عددي، يستخدم في التعريف فقط ، وعلى هذا .. استخدمت X في تعريفه .
- (٣) كلمة FILLER هي كلمة محجوزة، تستخدم في موقع اسم بيانات لحقل لن يجرى عليه تشغيل في جزء الإجراءات . وقد استخدمت هنا في تسمية حقول فارغة تفصل عناصر بيانات فعلية عند طباعتها أو عرضها على شاشة .
- (٤) تم تنقيح INVOICE- QUANTITY- PURCHASED للطباعة . وهو حقل من 3 خانات (2 ممثلتين بالحرف Z خانتين والثالث بالرمز 9) ، وتكون أول خانتين فارغتين إذا احتويتا على أصفار .
- (٥) تم تنقيح INVOICE- TOTAL- PRICE, INVOICE- UNIT- PRICE للطباعة . ويشمل هذا وضع فراغات بدلا من الأصفار الرائدة وطباعة العلامة العشرية الفعلية . لاحظ استخدام 9 بدلا من Z بعد " . " في PIC . وهذا لتأكيد ان العلامة العشرية تطبع دائما ، وتكون رموز التنقيط المحاطة بالرمز Z في PIC فراغات إذا ما كانت Z السابقة والتالية تمثل أصفارا رائدة .

رمز تنقيح الفاصلة

تمثل الفاصلة " ، " بايت واحداً يوضع فيه رمز الفاصلة . ومثل النقطة .. فظهر فاصلة بين اثنين Z تستبدلان بفراغ ، يجعل الفاصلة تستبدل بفراغ كذلك . كما ان الفاصلة تستبدل بفراغ كذلك اذا ما وجدت فراغات فقط على يسارها . لا يمكن استخدام الحقول العددية التي يوجد فاصلة في الصور الخاصة بها في الحسابات (وذلك بالرغم من أنها يمكن أن تستقبل حقولا عددية) .

مثال ه - ٢٨ :

PICTURE	Value	Prints As
ZZ,ZZZ.99	00010,05	bbbb10.05
ZZ,ZZZ.99	00123,45	bbb123.45
ZZ,ZZZ.99	01000,00	b1,000.00
ZZ,ZZZ.99	98765,43	98,765.43
ZZ,ZZZ.ZZ	00000,00	bbbbbbbbb
ZZ,ZZZ.ZZ	00000,03	bbbbbbbbb3
		(falsified)
ZZ,ZZZ.99	00000,03	bbbbbb.03
		(correct version of above)

علامة الدولار كرمز تنقيح

يمكن أن تستخدم علامة الدولار في الصورة لإدخال علامة دولار ثابتة fixed dollar sign ، تطبع على يسار أول موضع رقم مباشرة .

مثال ه - ٢٩ :

PICTURE	Value	Prints As
\$Z,ZZZ.99	0072,15	\$bbb72.15
\$Z,ZZZ.99	9876,54	\$9,876.54

يمكن أن تحدد علامة الدولار علامة دولار متحركة floating dollar sign تطبع على يسار أول رمز معنى significant character مباشرة (أول رمز معنى هو أول رقم معنى من العنصر العددي ، أو من الممكن أن يكون العلامة العشرية نفسها) . بهذا الاستخدام يكون هناك حاجة لأكثر من علامة دولار واحدة في PICTURE : تمثل العلامة الموجودة على أقصى اليسار موقعا لعلامة الدولار نفسها ، أما بقية علامات الدولار .. فتتمثل مواقع أرقام في الحقل .

مثال ه - ٣٠ :

	PICTURE	Value	Prints As
(a)	\$\$,\$\$\$99	0001,23	bbbb\$1.23
(b)	\$\$,\$\$\$99	0845,79	bb\$845.79
(c)	\$\$,\$\$\$99	9876,54	\$9,876.54
(d)	\$\$,\$\$\$99	12345,67	\$2,345.67
(e)	\$\$,\$\$\$99	-0000,05	bbbbbb\$.05

تعليقات :

من (أ) الى (ح) .. تتحرك علامة الدولار إلى أعلى رقم في القيمة المنقحة . (د) يوجد 6 مواقع أرقام فقط في -PIC
TURE (4 علامات دولار بعد علامة الدولار الأولى بالإضافة إلى 2 من رمز 9) . وعلى هذا .. القيمة المكونة من سبعة أرقام،
يحذف منها رقم من ناحية اليسار .

(هـ) لا يوجد تحديد لإشارة جبرية في PICTURE ، وتطبع على هذا القيمة كما لو كانت موجبة . وهنا أول رمز معنوى هو
العلامة العشرية وتنتقل علامة الدولار إليها .

تذكر أن علامة الدولار المتحركة تتطلب علامة دولار إضافية في PICTURE عن عدد مواقع الأرقام التى يجرى عليها
التنقيح . ولا يمكن استخدام العناصر التى لها علامة دولار في PIC الخاصة بها .

رموز تنقيح التحكم فى الإشارة DB , CR , + , -

تستخدم هذه الرموز أساسا فى تنقيح القيم العددية السالبة .

مثال ٥ - ٣١ :

يمكن استخدام رموز الصورة DB أو CR فى الناحية اليمنى من صورة العدد، لتحديد قيمة المدين (أو الدائن) السالبة .

	PICTURE	Value	Prints As
(a)	\$Z,ZZZ.99CR	0010.56	\$bbb10.56bb
(b)	\$Z,ZZZ.99CR	-1435.72	\$1,435.72CR
(c)	\$\$,\$\$\$99BDB	-3476.52	\$3,476.52 DB

تعليقات :

(أ) و (ب) اذا كانت القيمة سالبة .. فتطبع CR أو DB) وإلا فإنها تستبدل بفراغات .

(جـ) تتسبب B هنا فى وجود فراغ بين آخر رقم مطبوع ، و DB أو CR) .

رمزا تحكم الإشارة الآخران هما : + و - ويعملان بالتماثل ، إلا فى وجه واحد : تتسبب + فى طباعة إشارة العدد دائما
سواء (كان موجبا أو سالبا) بينما تتسبب - فى إظهار الإشارة عندما تكون القيمة المنقحة سالبة فقط . ويمكن استخدام رموز
الصورة هذه فى موقع رائد leading position (يسبق أول موقع لرقم مباشرة) أو فى موقع متخلف trailing posititon (بعد
موقع آخر رقم مباشرة) . وتستخدمان كذلك فى حالة ثابتة أو متحركة مثل علامة الدولار تماما .

مثال ٥ - ٣٢ :

	PICTURE	Value	Prints As
(a)	\$\$,\$\$\$99-	-2345.67	\$2,345.67-
(b)	\$\$,\$\$\$99-	0023.78	bbb\$23.78b
(c)	ZZZ.9+	-763.7	763.7-

(d)	-ZZ.9	23.7	b23.7
(e)	-ZZ.9	-00.7	-bb.7
(f)	+ZZ9	-005	-bb5
(g)	++++.9	0003.7	bbbb + 3.7
(h)	---9	0000	bbbbbb0
(i)	---9	-0100	bb - 100
(j)	++++9	0000	bbbb + 0

تعليقات :

- (أ) يتكرر استخدام إشارة السالب المتخلفة في تقارير الأعمال لإظهار الأعداد السالبة .
 (ب) ترك موقع إشارة السالب المتخلفة فارغا للقيمة الموجبة .
 (ج) يحتوى موقع إشارة الموجب المتخلفة على إشارة القيمة .
 (د) حل محل إشارة سالب الرائدة الثابتة فراغ للقيمة غير السالبة .
 (هـ) تطبع إشارة سالب الثابتة الرائدة دائما في أول موقع .
 (و) أشارة الموجب الثابتة الرائدة تضع الإشارة في أول موقع دائما .
 (ر) أشارة الموجب المتحركة تنتقل إلى أول رمز معنوى .
 (ز) أشارة السالب المتحركة تطبع إذا كانت القيمة سالبة فقط ، فيجبر إنهاء PIC بالرمز 9 ، 0 على طباعته .
 (ع) تطبع إشارة السالب المتحركة عندما تكون القيمة سالبة .
 (غ) الصفر أعتبر موجبا .
 في أعداد تقارير الأعمال ، عادة ما تعالج تحركات الإشارة بإشارة سالب متخلفة أو برمز CD أو رمز DB متخلف .
 وكما هو الحال مع رموز التنقيح الأخرى لا يمكن استخدام الحقول الموجود في PIC الخاص بها رموز تحكم في الإشارة في الحسابات .

النجمة كرمز تنقيح

تشير حماية الشيكات check protection إلى عمليات ملأ مواقع الأصفار الرائدة في العدد بنجوم، بحيث يمكن حماية القيمة المدونة في الشيك من التزوير : 23.45 ***\$. وتنفذ حماية الشيكات برمز النجمة في الصورة .

مثال ه - ٣٣ :

	PICTURE	Value	Prints As
(a)	\$*,***.99	12345.67	\$2,345.67
(b)	\$*,***,***.99	0001234.56	\$****1,234.56
(c)	\$*,***.**	0000.00	\$*****.

تعليقات :

- (أ) يمكن لعدد 4 نجوم و 2 من الرمز 9 أن يتسعوا لستة أرقام ، ويحدث حذف من ناحية اليسار .
- (ب) استبدلت الأصفار الرائدة والفواصل الرائدة بنجوم ، وتبقى الأرقام المعنوية والفواصل المعنوية .
- (جـ) تطبع العلامة العشرية دائما عندما تستخدم حماية الشيكات حتى اذا كانت القيمة مساوية للصفر .

الصفر كرمز تنقيح

يستخدم رمز الصورة 0 فى حجز موقع داخل النتيجة المنقحة، يطبع فيه صفر .

مثال ٥ - ٢٤ :

أحد الاستخدامات التقليدية للصفر كرمز تنقيح ، هو تقديم صفر من الستات بحيث يمكن طباعة العناصر - التى تحفظ كدولارات فى ذاكرة الكمبيوتر - كدولارات وستات .

PICTURE	Value	Prints As
\$ZZ,ZZZ.00	01205	\$b 1,205.00
\$ZZ,ZZZ.00	00002	\$bbbbbb2.00
\$\$\$,\$\$\$0.00	00002	bbbbbb\$2.00

الشرطة المائلة كرمز تنقيح

يستخدم رمز الصورة " / " فى حفظ بايت فى النتيجة المنقحة يمسك دائما بشرطة مائلة . تفيد الشرطة المائلة خاصة عند تنقيح إعداد تمثيل تواريخ (مثل يمكن أن يتقح 102383 ليأخذ الشكل 10/23/83 باستخدام PIC 99/99/99 .

رمز التنقيح P

تستخدم سلسلة من الرمز P لتحديد موقع علامة عشرية افتراضية ، عندما يقع خارج مدى الأرقام المحفوظة فى الذاكرة . لا يعد P رمز تنقيح، ويستخدم فى تعريف حقول عددية تستخدم فى الحسابات . ولا يشغل P أى موقع تخزين إضافي فى الحقل، كما أنه موقع يحدد موقع علامة عشرية افتراضية . ويمكن أن يظهر P فى النهاية اليسرى فقط أو النهاية اليمنى فقط من الحقل العددي . والرموز الوحيدة التى يمكن أن تسبق P على اليسار هي V ، S ، والرموز الوحيدة التى يمكن أن تتبع P على اليمين هي V .

مثال ٥ - ٢٥ :

	PICTURE	Value in Memory	Value Used in Computation
(a)	PP99	12	.0012
(b)	99PP	12	1200.
(c)	PP99PP	—	—
(d)	SPPP999	735	+ .000735
(e)	SVPPP999	735	+ .000735
(f)	SPP999	1234	+ .00234
(g)	S9(3)P(4)	538	+5380000.
(h)	SP99	03	+ .003

تعليقات :

- (أ) و (ب) و (ر) تقع العلامة العشرية الافتراضية بعد عدد المواقع التي تمثلها P من ناحية اليسار .
 (ج) غير صحيح : لا يمكن أن تظهر P في كل من النهايتين .
 (د) S يمكن أن تسبق P من ناحية اليسار .
 (هـ) V زائدة هنا ، لأن العلامة العشرية الافتراضية تم تثبيتها فعلا على يسار P الموجودة في أقصى اليسار .
 (و) إذا كانت للقيمة في الذاكرة أرقام أكثر عن عدد رمز 9 في PIC .. تحذف أرقام من ناحية اليسار ، وتحدد العلامة العشرية الافتراضية في القيمة المحذوفة بواسطة P .
 (ر) لايعنى شيئا لعمل P ان الرقم الموجود على أقصى اليسار صفر .

٥ - ١١ جزء الاستخدام (شكل ٥ . ٧ - السطر رقم 3)

عادة مايحذف الجزء الذى يحدد التمثيل الداخلى المستخدم لعنصر البيانات ، من وصف البيانات حيث إن استخدام DIS-PLAY التقليدى ، هو الأكثر مناسبة فى استخدامه .

الاستخدام هو للعرض

تحت DISPLAY يمثل عنصر البيانات بشفرة رموز نظام الكمبيوتر (سواء كانت EBCDIC أم ASCII) ، والتي يمثل فيها الرمز الحرفى عددي بشفرة ثنائية تحتاج إلى بايت واحد من الذاكرة . تكون DISPLAY إجبارية إذا كان عنصر البيانات (١) حرفياً عددياً PIC X (٢) حرفياً PIC A (٣) أى نوع للحقل فى سجل ملف بطاقة (سواء كان مدخلات أو مخرجات) . (٤) أى نوع للحقل فى سجل مخرجات على طابع (٥) أى نوع للحقل يكون مدخلات أم مخرجات على شاشة أو نهاية طرفية تنتج نسخاً دائمة (٦) حقلاً عددياً تشمل صورته رمزاً واحداً أو أكثر من رموز التنقيح (٧) أى نوع من حقل المدخلات من ملف كان DISPIAY فعلا . ويوصى باستخدام DISPLAY إذا كان عنصر البيانات حقلاً عددياً مستخدماً فى التعريف ، وليس فى الحسابات (يمكن أن يكون فى مخزن العمل أو فى أى نوع من أنواع وحدات التخزين المساعد للمدخلات والمخرجات) .

مثال ٥ - ٣٦ :

بعد كل حقل مكتوب الأسباب التى من أجلها يجب أن تظهر DISPLAY باستخدام الأرقام المذكورة عليه .

```
01 INPUT-TIME-CARD-RECORD.
05 TIME-CARD-ID          PIC X(5)      USAGE IS DISPLAY. (1, 3, 7)
05 TIME-CARD-NAME        PIC A(20)     USAGE DISPLAY.    (2, 3, 7)
05 TIME-CARD-HOURS       PIC S9(2)V9.   (3, 7)
```

الاستخدام يكون للحسابات (COMP)

COMP هو الاستخدام النمطى لكويل ANS للعناصر العددية التى تستخدم فى الحسابات . ويتغير تمثيل البيانات الدقيقة المستخدم لبيانات COMP من كمبيوتر لآخر (وعادة ماتكون إحدى صيغ مكمل الأزواج الثنائية - binary- twos-

complement . وبالرغم من أن الكمبيوتر يمكن أن يؤدي حسابات على أعداد COMP بكفاءة أعلى من أعداد DISPLAY ، إلا إنه يوصى بالنسبة للمبرمج الجديد أن يتذكر مايلي : لاستخدام COMP إذا كان عنصر البيانات (١) محتلاً DISPLAY (٢) غير عددي أو إنه منقحاً (٢) حقلاً مصاحباً لوحدة سجل وحدة أو نهاية لانتاج نسخة دائمة أو شاشة (٤) عددياً ، ولكنه مستخدم في التعريف فقط (استخدام PIC X DISPLAY بدلا من ذلك) .

استخدم COMP إذا كان عنصر البيانات (١) حقلاً عددياً في سجل مدخلات من ملف شريط أو قرصاً وهو COMP فعلاً (ب) حقلاً عددياً مستخدماً في الحسابات وهو حقلاً في سجل يكون مخرجات في ملف شريط أو قرصاً يصبح بعد ذلك مدخلات ويستخدم في حسابات لاحقة (جـ) في مخزن العمل (ليس حقلاً سجل ملف) ويكون عددياً غير منقح، ويستخدم في الحسابات .

مثال ٥ - ٢٧ :

```
01 DISK-OUTPUT-RECORD.
05 DISK-CUSTOMER-NUMBER PIC X(6).
05 DISK-CUSTOMER-NAME   PIC X(30).
05 DISK-LAWYER-NUMBER   PIC X(3).
05 DISK-JOB-NUMBER       PIC X(4).
05 DISK-JOB-HOURS        PIC S9(3)V9 COMP.
05 DISK-JOB-RATE         PIC S9(3)V99 COMP.
```

رقم العميل، ورقم المحامي، ورقم العمل... كلها أرقام مستخدمة في التعريف فقط ، وعلى هذا فهي معرفة باستخدام PIC X DISPLAY (تذكر أن DISPLAY هو الخيار التقليدي) . رقم العميل من النوع الحرفي عددي ، ومن ثم فلا بد أن يوصف على هيئة PIC X DISPLAY . عدد ساعات العمل ومعدل الأجر في الساعة هما حقلاً عددياً يجري عليهما حسابات . وعلى هذا .. فمن الكفاءة الأكثر تخزينها على قرص في صورة COMP . وفيما بعد... يصبحان مدخلات في عناصر بيانات COMP ، ويستخدمان في حساب قيمة فاتورة العميل .. إلخ .

مثال ٥ - ٢٨ :

```
.....
05 ORIGINAL-PRICE      PIC S9(3)V99.
05 DISCOUNT           PIC S9(3)V99.
05 DISCOUNTED-PRICE   PIC S9(3)V99.
.....
PROCEDURE DIVISION.
.....
SUBTRACT DISCOUNT FROM ORIGINAL-PRICE GIVING
DISCOUNTED-PRICE
```

حيث أن كل عناصر البيانات الثلاث من نوع DISPLAY فإن المترجم يدرك ضرورة تحويلها إلى COMP أو (COMP-3) بغرض أداء الحسابات . وبعد ذلك تحول النتائج مرة أخرى إلى DISPLAY . ويمكن أن تكون الترجمة التقليدية للغة الآلة لعملية الطرح كما يلي :

- (١) حول ORIGINAL- PRICE إلى صيغة COMP ، وضع النتيجة في موقع مؤقت من الذاكرة T1 .
 (٢) حول DISCOUNT إلى صيغة COMP وضع النتيجة في موقع مؤقت من الذاكرة T2 .
 (٣) أطرح محتويات موقع الذاكرة T2 من محتويات موقع الذاكرة T1 ، وضع الفرق في موقع مؤقت من الذاكرة T3 .
 (٤) حول محتويات موقع الذاكرة المؤقت T3 إلى صيغة DISPLAY ، وضع النتيجة في DISCOUNTED- PRICE .
 إذا عرف عناصر البيانات الثلاثة بأنها من نوع USAGE COMP .. فيمكن أن تصبح الترجمة اقتصادية أكثر :
 (١) أطرح محتويات DICOOUNT من محتويات ORIGINAL- PRICE وضع الفرق في الموقع DISCOUNTED- PRICE

الاستخدام من نوع الحسابات (COMP-3)

COMP-3 هو أتساع لكويل ANS المتاح على بعض ، و ليس كل ، نظم الكمبيوتر ، وهو يشبه COMP في قدرة وحدة التشغيل المركزية على عمل حسابات بكل من أعداد COMP ، COMP-3 .
 كما أن كلاً من الشفرتين تحتاج إلى ذاكرة أقل من تمثيل قيمة عددية معينة، مستخدمة في حسابات من أنواع حسابات الأعمال (أفحص دليل المورد المتاح لديك) . استثناء : لاستخدام COMP-3 للدالة subscripts (انظر الفصل العاشر) .
 مثال ٥ - ٣٩ :

استبدال - مثال ٥ - ٣٨ - PIC S9 (3) V99 بـ PIC S9 (3) V99 COMP-3 يعطى برنامج هدف أكثر كفاءة على كمبيوتر من نوع IBM-370 . استخدام COMP لعناصر البيانات الثلاثة يعطى ترجمة أقل كفاءة مباشرة اما استخدام DI- SPLAY (كما في مثال ٥ - ٣٨) فيعطى أقل كفاءة ترجمة .

٥ - ١٢ جزء الفراغ (شكل ٥ . ٧ - السطر رقم 4)

يتسبب جزء BLANK WHEN ZERO في وضع أماكن فراغة في عنصر البيانات عندما ينتقل إليه أصفاراً . يمكن استخدام هذا الجزء مع العناصر العددية التي لا يستخدم معها حماية للشيكات فقط . ويعتبر عنصر البيانات الذي يصاحبه BLANK WHEN ZERO ، بأنه عنصر تنقيح تلقائي ، وعلى هذا .. يجب أن يكون USAGE له من نوع DISPLAY .

مثال ٥ - ٤٠ :

```
01 ACCOUNTS-RECEIVABLE-LINE.
05 A-R-CUSTOMER-NAME PIC X(30).
05 FILLER PIC X(5).
05 A-R-TYPE-PAYMENT PIC XB.
05 FILLER PIC X(4).
05 A-R-DAYS-PAST-DUE PIC ZZ9 BLANK WHEN ZERO.
05 FILLER PIC X(7).
05 A-R-AMOUNT-PAST-DUE PIC $$$$.99 BLANK WHEN ZERO.
```

إذا نقلت القيمة صفر إلى A- R- DAYS- PAST- DUE فإنه يطبع على هيئة bbbb (بدلاً من طباعته على هيئة bb0). وإذا نقل صفر في A- R- AMOUNT- PAST- DUE فإنه يطبع على هيئة bbbbbbbbbb (بدلاً من طباعته على هيئة bbbbbb\$.00). النتيجة هي طباعة القيم غير الصفرية فقط في التقرير ، وهذا يركز انتباه المستفيد على العملاء الذين عليهم ديون .

مثال ٥ - ٤١ :

يمكن أن يظهر التقرير المطبوع بواسطة مثال ٥ - ٤٠ ، بدون استخدام BLANK WHEN ZERO ، على النحو التالي :

NAME	TYPE PAYMENT	DAYS PAST	AMOUNT PAST
FRANK	A 7	0	\$.00
ACE	A 7	0	\$.00
CARTER	B 3	5	\$25.32
SMITH	A 7	0	\$.00
PERELMAN	B 5	63	\$587.00
CONVERSE	A 7	0	\$.00
GETZ	B 3	0	\$.00

لاحظ كيف تتطلب البيانات المعنوية انتباهاً آمناً، يمكن أن يضيع في زحام الأصفار .

٥ - ١٣ جزء التضبيط (شكل ٥ . ٧ - السطر رقم 5)

يمكن استخدام جزء التضبيط JUSTIFIED مع العناصر الحرفية عدديّة (PIC X)، أو العدديّة (PIC A) فقط . عادة ، وبصورة تقليدية .. تضبط البيانات الحرفية عدديّة والحرفية من ناحية اليسار left-justified في الجمل ، ويتسبب هذا الجزء في ضبطها من ناحية اليمين right-justified ، ونائراً ما يستخدم هذا الجزء .

مثال ٥ - ٤٢ :

05 FIELD-A PIC X(5).
05 FIELD-B PIC X(5) JUSTIFIED RIGHT.

افرض أن القيمة CAT نقلت إلى كل من FIELD-A و FIELD-B . عند ذلك ينتج مايلي :

Field	Prints As
FIELD-A	CATbb
FIELD-B	bbCAT

مثال ٥ - ٤٣ :

05 FIELD-A PIC X(3).
05 FIELD-B PIC X(3) JUSTIFIED RIGHT.

افترض أن القيمة SMILES نقلت إلى كل من : FIELD-A و FIELD-B . وعند ذلك ينتج مايلي :

	Field	Prints As
(a)	FIELD-A	SM1
(b)	FIELD-B	LES

تعليقات : (١) إذا كانت القيمة أطول من الحقل فتحذف العناصر المعرفة باستخدام PIC A , PIC X من نفس الناحية التي يحدث فيها ملء فراغات إذا كان طول القيمة أقصر من الحقل . وحيث أن التضييق التقليدي يكون من ناحية اليسار يحدث الحذف من ناحية اليمين وتنتقل الثلاثة رموز الموجودة في أقصى اليسار إلى الحقل .

(ب) يتسبب جزء JUSTIFIED في الضبط من ناحية اليمين . وعلى هذا تنتقل الثلاثة الرموز الموجودة في أقصى اليمين من القيمة إلى الحقل المعروف بواسطة (3) PICX .

٥ - ١٤ جزء الحثوث (شكل ه . ٧ - السطر رقم 6)

يسمح جزء الحثوث OCCURS للمبرمج بإنتاج هيكل بيانات ، عادة مايسمى بجدول table ، أو منظومة array ، ونناقش معالجة الجداول في الفصل العاشر .

مثال ه - ١٤ :

01	ITEM-DESCRIPTION-RECORD.		
05	ITEM-NUMBER	PIC X(7).	
05	ITEM-WAREHOUSE-BIN	PIC X(2).	
05	ITEM-ON-HAND	PIC S9(5)	COMP-3.
05	ITEM-SUPPLIERS	OCCURS	10 TIMES.
10	ITEM-SUPPLIER-NAME	PIC X(20).	
10	ITEM-SUPPLIER-ID	PIC X(5).	
10	ITEM-SUPPLIER-COST	PIC S9(5)V99	COMP-3.
05	ITEM-PRICE	PIC S9(5)V99	COMP-3.

تتكرر مجموعة العناصر ITEM- SUPPLIERS عدد 10 مرات في السجل ، مكونة بذلك جدولا table . ويجب أن يستخدم المبرمج دليلا subscript أو فهرسا index في جزء الإجراءات يحدد محتوى الجدول الذي يجرى تشغيله عليه .

MOVE 'NUTS AND BOLTS, INC.' TO ITEM-SUPPLIER-NAME (WHICH-SUPPLIER)

للعنصر العددي WHICH- SUPPLIER قيمة تقع بين 1 ، 10 ويحدد العناصر العشرة التي يجرى عليها تشغيل . يجب أن تعرف مثل هذه الادلة باستخدام USAGE COMP .

05 WHICH-SUPPLIER PIC S9(2) 'COMP.

٥ - ١٥ جزء الإشارة

يمكن استخدام جزء الإشارة SIGN للعناصر العددية المستخدم في تعريفها S ، ولها USAGE DISPLAY فقط، والبدائل التالية ممكنة .

- SIGN IS TRAILING

إشارة العملية هي جزء من التمثيل الداخلي للرقم الموجود على أقصى اليمين ولا تتطلب أى موقع تخزين إضافي . وهذا هو الوضع التقليدي لكويل IBM OS/VS .

- SIGN IS LEADING

إشارة العملية هي جزء من التمثيل الداخلي للرقم الموجود على أقصى اليسار . وهو لا يتطلب أى موقع تخزين إضافي .

- SIGN IS TRAILING SEPARATE CHARACTER

تضاف إشارة السالب أو الموجب في التمثيل الداخلي للعدد ، وذلك باستخدام شفرة EBCDIC أو شفرة ASCII . في هذه الحالة لا تمثل S الموجودة في PIC بايت إضافياً من التخزين للعنصر . وإذا لم توجد الإشارة في القيمة الموضوعة في عنصر البيانات هذا أثناء تنفيذ البرنامج ينتهي البرنامج نهاية غير طبيعية .

- SIGN IS LEADING SEPARATE CHARACTER

يسبق تمثيل شفرة EBCDIC أو شفرة ASCII لإشارة السالب أو الموجب، التمثيل الداخلي للعدد . ويمثل رمز الصورة S مرة أخرى بايت إضافياً لموقع تخزين مطلوب . وتناظر الإشارة الرائدة المنفصلة LEADING SEPARATE الطريقة التي يكتب بها الإنسان الأعداد السالبة . فإذا لم توجد الإشارة في القيمة المنقولة إلى هذا العنصر أثناء التنفيذ ، فينتهي البرنامج نهاية غير طبيعية .

DATA-ITEM PIC S9(3)

مثال ٥ - ٤٥ :

بالقيمة +123 ، يبين شكل (٥ - ٨) التمثيل الداخلي لاسم البيانات DATA-ITEM ، لعدد من أجزاء SIGN .

SIGN IS TRAILING:

1	2	3	+
---	---	---	---

 (3 bytes)

SIGN IS LEADING:

+	1	2	3
---	---	---	---

 (3 bytes)

SIGN IS TRAILING SEPARATE CHARACTER:

1	2	3	+
---	---	---	---

 (4 bytes)

SIGN IS LEADING SEPARATE CHARACTER:

+	1	2	3
---	---	---	---

 (4 bytes)

شكل (٥ - ٨)

ينتج عن البديلين LEADING , SEPARATE CHARACTER برنامج هدف أقل كفاءة بالنسبة لأجهزة كمبيوتر IBM 370 . من الأفضل بصفة عامة أن تحذف SIGN ويقبل التنفيذ التقليدي لإشارة العملية على أى نظام كمبيوتر .

٥ - ١٦ جزء التوافق (شكل ٥ . ٧ - السطر رقم 8)

يمكن، على بعض أجهزة مثل IBM 370 ، معالجة العناصر العددية المعرفة باستخدام USAGE COMP بكفاءة أكثر إذا ما ضبطت aligned مواقع الذاكرة المحددة للعنصر على الحد boundary المناسب . إذ يمكن تشغيل عنصر معرف باستخدام COMP يشغل 2 بايت بكفاءة أعلى إذا كان عنوان أول بايت يقبل القسمة على 2 ، والحقل المعرف باستخدام COMP ويشغل 4 بايت، بكفاءة أعلى إذا مابء ببایت عنوان يقبل القسمة على 4 .

جزء التوافق SYNCHRONIZED (SYNC) يتسبب في أن يقوم المترجم بتحديد موقع العناصر المعرفة باستخدام COMP بحيث أنها تكون مضبوطة علي الحد الأكثر كفاءة . قد يتطلب هذا إدخال بعض البايت الراكدة slacle bytes بين الحقل السابق في السجل والحقل الذي يضبط ، وتقدم هذه تلقائيا بواسطة المترجم .

مثال ٥ - ٤٦ :

بمعرفة وصف السجل التالي :

Length in Bytes

	01	ORDER-RECORD.	
4	05	ORDER-NUMBER	PIC X(4).
5	05	ORDER-CUSTOMER-ID	PIC X(5).
6	05	ORDER-ITEM-NO	PIC X(6).
2	05	ORDER-QUANTITY	PIC S9(4) COMP SYNC.
4	05	ORDER-UNIT-PRICE	PIC S9(5)V99 COMP SYNC.

يمكن أن يكون تخطيط الذاكرة للسجل ORDER- RECORD كما يلي (للبايت الموجود في أقصى اليسار .. العنوان 0).

4 bytes	5 bytes	6 bytes	1 byte	2 bytes	2 bytes	4 bytes
Order #	Customer ID	Item #	slack	Quantity	slack	Unit Price

نظرا لأن البايت الراكدة هي مخازن مفقودة ، ويمكن أن تقود إلى أخطاء في تحديد أطوال السجلات ، يوصى باستخدام COMP SYNC كدلائل فقط على أجهزة كمبيوتر IBM-370 .

مثال ٥ - ٤٧ :

من الأفضل تعريف WHICH-SUPPLIER في مثال ٥ - ٤٤، كما يلي :

05 WHICH-SUPPLIER PIC S9(2) COMP SYNC.

يجب ملاحظة انه في كويل IBM OS/VS تهمل RIGHT, LEFT الموجودة في SYNCHRONIZED.

٥ - ١٧ جزء إعادة التعريف (شكل ٥ . ٧ - السطر رقم 1)

يستخدم جزء إعادة التعريف REDEFINES في تقديم وصف بديل لعنصر بيانات فردي أو مجموعة عناصر . وعند استخدام REDEFINES يجب أن يكون أول جزء يتبع رقم المستوى واسم البيانات .

level-number data-name-1 REDEFINES data-name-2

Data-name-1 هو الاسم البديل لمنطقة بيانات ، بينما Data-name-2 هو اسمه السابق . ويجب أن يكون رقم المستوى المحدد لـ Data-name-1 مثل رقم مستوى Data-name-2 تماما .

مثال ٥ - ٤٨ :

FD CUSTOMER-MASTER-FILE
RECORD CONTAINS 105 CHARACTERS
LABEL RECORDS ARE STANDARD

01 CUSTOMER-MASTER-RECORD.

05	CUSTOMER-ID	PIC X(6).
05	CUSTOMER-NAME	PIC X(30).
05	CUSTOMER-ADDRESS	PIC X(30).
05	CUSTOMER-REGULAR-DATA.	
10	CUSTOMER-BILLING-TYPE	PIC XX.
10	CUSTOMER-CREDIT-LIMIT	PIC S9(5)V99 COMP-3.
10	CUSTOMER-WEIGHT-LIMIT	PIC S9(3)V9 COMP-3.
05	CUSTOMER-PREFERRED-DATA	
	REDEFINES CUSTOMER-REGULAR-DATA.	
10	PREFERRED-DISCOUNT-RATE	PIC S9(5)V99 COMP-3.
10	PREFERRED-LOCATION-CODE	PIC X(3).
10	PREFERRED-SHIPPING-MODE	PIC X(3).
10	PREFERRED-BILLING-TYPE	PIC X.
05	CUSTOMER-CITY-STATE-ZIP.	
10	CUSTOMER-CITY	PIC X(19).
10	CUSTOMER-STATE	PIC X(2).
10	CUSTOMER-ZIP	PIC X(9).

حيث إن الملف محفوظ على قرص ، فيمكن استخدام COMP-3 للعناصر العددية التي يجرى عليها حسابات . ولا يوفر ذلك من المكان على القرص فقط بل تنتج عنه شفرة هدف أكثر كفاءة كذلك . يتسبب جزء REDEFINES في أن يشغل كل من CUSTOMER- REGULAR- DATA و CUSTOMER- PREFERRED- DATA نفس مواقع الذاكرة تماما (حيث إن هذا هو FD) فتكون المواقع جزءا من السجل المنطقي النشط حاليا في الذاكرة الاحتياطية لل ملف) . وعلى هذا توجد منطقة من 9 بايت من السجل المنطقي النشط حاليا ، يمكن أن يكون لها تفسيران مختلفان تماما (شكل ٥ - ٩) .

ويرجع إلى المبرمج ما إذا كان يريد العمل بأسماء بيانات تتفق اتفاقاً صحيحاً مع محتويات الذاكرة في أى نقطة من نقاط تنفيذ البرنامج . مثال ذلك .. يمكن أن يكون لكل العملاء المفضلين CUSTOMER- ID ينتهى بصفر . بعد قراءة CUSTOM- ER- MASTER- RECORD .. يمكن أن يختبر البرنامج آخر رمز من CUSTOMER- ID . إذا لم يكن صفراً فسوف يجرى تشغيل لعناصر CUSTOMER- REGULAR- DATA أما إذا كان صفراً .. فسوف يجرى تشغيل CUSTOM- ER- PREFERRED- DATA

CUSTOMER-PREFERRED-DATA

Discount Rate Pic SV99 COMP-3 (2 bytes)	Location Code Pic X(3) (3 bytes)	Shipping Mode Pic X(3) (3 bytes)	Billing Type Pic X (1 byte)
part of currently active logical record in buffer			
Billing Type PIC XX (2 bytes)	Credit Limit PIC S9(5)V99 COMP-3 (4 bytes)	Weight Limit PIC S9(3)V9 COMP-3 (3 bytes)	

CUSTOMER-REGULAR-DATA

شكل (٥ - ٩)

لاحظ وجود الحرية الكاملة في إعادة تعريف REDFINES منطقة ، بالنسبة الى USAGE, PICTURE وغيرها . القيد الوحيد هو أن طول العنصر المعاد تعريفه يجب أن يتساوى مع طول العنصر الذي يعيد تعريفه . لاحظ كذلك REDFINES يجب أن تؤخذ في الحسبان عند تحديد طول السجل ، فطول CUSTOMER- MASTER- DATA هو 105 بايت وليس 104 .

مثال ٥ - ٤٩ :

في برنامج مثال ٥ . ١٦ العبارة التالية :

```
01 MAILING-LABEL-CITY-STATE-REC
   REDEFINES MAILING-LABEL-NAME-STREET-REC.
```

يمكن أن تتسبب في خطأ تكويني (ويمكن أن تكون غير ضرورية ، حيث يعيد العنصران على المستوى 01 بعضهما البعض ضمناً implicitly redefine) . يجب ألا يعاد تعريف العناصر في مستوى 01 في قسم الملفات صراحة . ولكن يمكن أن يعاد تعريفها في مخزن العمل .

مثال ٥ - ٥٠ :

عادة يكون اختبار حقول المدخلات مساعداً في التصحيح (التأكد من مهمة validation المدخلات) وذلك بجعله قادراً على الإشارة إلى حقل عددي معين بأن له PIC9, PICX . ويسمح REDEFINE المبرمج بالحصول عليها بالطريقتين :

```
.....
05 NUMERIC-INPUT-FIELD PIC S9(3)V99.
05 NUMERIC-INPUT-FIELD-AS-X
   REDEFINES NUMERIC-INPUT-FIELD PIC X(5).
```

٥ - ١٨ تحديد طول عنصر البيانات

كمية الذاكرة اللازمة لتخزين عنصر معين تتحدد بواسطة PICTURE و USAGE ؛ حيث يعتبر طول مجموعة العناصر هو مجموع أطوال عناصرها الفردية ، فيكفي معرفة كيفية إيجاد طول أى عنصر فردي .

- DISPLAY USAGE : كل رمز صورة (باستثناء V, S, P) يتطلب بايت واحداً من التخزين .
- COMP USAGE : حيث إنه لا يمكن تنقيح عناصر COMP .. فان P, 9, V, S هي رموز الصورة الوحيدة المسموح بها . وكما هو الحال مع عناصر DISPLAY فلا تسهم P, V, S في كمية الذاكرة اللازمة . واسوء الحظ ، فإن عدد الرموز في COMP PICTURE لا يحدد بنفسه الطول . وبالنسبة لأجهزة كمبيوتر IBM 370 فإن :

Number of "9"s in PIC	Length of Item in Bytes
1 to 4	2
5 to 9	4
10 to 18	8

- COMP-3 USAGE : الرموز P, 9, V, S هي رموز الصورة الوحيدة المسموح بها . وكما هو الحال مع COMP .. فإن طول عناصر COMP-3 يعتمد على عدد الأرقام في العدد (عدد 9 في الصورة) . إذا كان هناك K من 9 (حيث $1 \leq k \leq 18$) في صورة COMP-3 .. فإن عدد البايت في العنصر يكون $1 + [K/2]$. حيث $[K/2]$ تعني أكبر رقم أقل من أو يساوي K/2 (أى حذف أى كسر) .

مثال ٥ - ٥١ :

01	DISK-INVENTORY-RECORD.		
05	INVENTORY-PART-NUMBER	PIC X(8).	
05	INVENTORY-DESCRIPTION	PIC X(30).	
05	INVENTORY-ON-HAND	PIC S9(5)	COMP.
05	INVENTORY-ON-ORDER	PIC S9(4)	COMP.
05	INVENTORY-DISCOUNT	PIC S9(3)V99	COMP.
05	INVENTORY-PRICE-WE-CHARGE	PIC S9(5)V99	COMP.
05	INVENTORY-ORDERED-DATE.		
10	INVENTORY-ORDERED-MONTH	PIC XX.	
10	INVENTORY-ORDERED-DAY	PIC XX.	
10	INVENTORY-ORDERED-YEAR	PIC XX.	
05	INVENTORY-PERCENT-SCRAPPED	PIC SVP9	COMP.

يجب أن يكون لدينا مايلي، على أجهزة IBM 370 :

<i>Data Name</i>	<i>Length in Bytes</i>	<i>Comment</i>
DISK-INVENTORY-RECORD	60	record length is sum of field lengths
INVENTORY-PART-NUMBER	8	one byte for each PIC X
INVENTORY-DESCRIPTION	30	one byte for each PIC X
INVENTORY-ON-HAND	4	5 to 9 digits COMP: 4 bytes
INVENTORY-ON-ORDER	2	1 to 4 digits COMP: 2 bytes
INVENTORY-DISCOUNT	4	5 to 9 digits COMP: 4 bytes
INVENTORY-PRICE-WE-CHARGE	4	5 to 9 digits COMP: 4 bytes
INVENTORY-ORDERED-DATE	6	field length is sum of subfield lengths
INVENTORY-ORDERED-MONTH	2	one byte for each PIC X
INVENTORY-ORDERED-DAY	2	one byte for each PIC X
INVENTORY-ORDERED-YEAR	2	one byte for each PIC X
INVENTORY-PERCENT-SCRAPPED	2	1 to 4 digits COMP: 2 bytes (note only one 9 in PIC)

مع استخدام COMP-3 ، بدلا من COMP ، تحدث التغييرات التالية في أطوال الحقول السابقة .

<i>Data Name</i>	<i>Length in Bytes</i>	<i>Comment</i>
DISK-INVENTORY-RECORD	58	2 bytes saved by using COMP-3
INVENTORY-ON-HAND	$[5/2] + 1 = 3$	saves 1 byte versus COMP; there are five "9"s in PICTURE
INVENTORY-ON-ORDER	$[4/2] + 1 = 3$	takes 1 extra byte versus COMP; there are four "9"s in PICTURE
INVENTORY-DISCOUNT	$[5/2] + 1 = 3$	saves 1 byte versus COMP
INVENTORY-PRICE-WE-CHARGE	$[7/2] + 1 = 4$	same as COMP
INVENTORY-PERCENT-SCRAPPED	$[1/2] + 1 = 1$	saves 1 byte versus COMP

٥ - ١٩ قسم مخزن العمل

يسمح قسم مخزن العمل من جزء البيانات المبرمج بتعريف مخازن رئيسية لمناطق بيانات ، يحتاجها البرنامج ولا تكون جزءا من أى سجل من السجلات المنطقية التى يجرى عليها البرنامج تشغيل (انظر المشكلة ٢١ - الفصل الثانى) . ويمكن وصف بيانات قسم مخزن العمل بطريقة تشبه تماما وضعها فى قسم الملفات ، مع الاختلافات التالية :

- (١) حيث ان حقول مخزن العمل ليست جزءا من سجلات منطقية .. فلا توجد محتويات FD فى قسم مخزن العمل .
- (٢) هناك رقم مستوى جديد 77، يستخدم فى تعريف حقول مخزن العمل التى لا تكون جزءا من هيكل سجل .
- (٣) هناك جزء جديد لوصف البيانات (VALUE) ، يحدد القيمة التى يجب أن يأخذها عنصر قسم مخزن العمل، عند بدء تنفيذ البرنامج .

المستوى 77

بينما تكون كل الحقول فى قسم الملفات إما سجلات منطقية أو أجزاء من سجلات منطقية ، فالعديد من حقول قسم مخزن العمل عبارة عن عناصر مستقلة independent items ، ليست لها علاقة بأى حقول أخرى فى مخزن العمل . ويمكن أن يعطى لكل عنصر من العناصر المستقلة رقم المستوى 77 مع وضعها فى بداية مخزن العمل . إلا أننا نوصى ، كلما كان ذلك ممكنا ، أن تجمع العناصر المستقلة فى سجلات مخزن عمل ، لها المستوى 01 (كما فى حالة قسم الملفات) . وهذا يسهل من تصحيح البرنامج وصيانته .

مثال ٥ - ٥٢ :

قارن الصيغتين التاليتين من قسم مخزن العمل :

•	77	NUMBER-OF-CHECKS-PRINTED	PIC S9(5)	COMP-3.
	77	NUMBER-OF-OVERTIME-WORKERS	PIC S9(5)	COMP-3.
	77	TAX-TABLE-SUBSCRIPT	PIC S9(4)	COMP SYNC.
	77	ALL-RECORDS-PROCESSED-SW	PIC X.	
	77	DEDUCTION-TABLE-SUBSCRIPT	PIC S9(4)	COMP SYNC.
	77	BELONGS-CREDIT-UNION-SW	PIC X.	
	01	PAYROLL-REPORT-LINE-AREA.		
	05	PAYROLL-EMPLOYEE-ID	PIC X(5).	
	05	FILLER	PIC X(3)	VALUE SPACES.
	05	PAYROLL-EMPLOYEE-NAME	PIC X(20).	
	05	FILLER	PIC X(4)	VALUE SPACES.
	05	PAYROLL-GROSS-PAY	PIC \$\$\$,\$\$\$,99.	
•	01	WORKING-COUNTERS.		
	05	WORKING-NUMBER-CHECKS-PRINTED	PIC S9(5)	COMP-3.
	05	WORKING-NUM-OVERTIME-WORKERS	PIC S9(5)	COMP-3.
	01	WORKING-PROGRAM-SWITCHES.		
	05	ALL-RECORDS-PROCESSED-SW	PIC X.	
	05	BELONGS-CREDIT-UNION-SW	PIC X.	

01 WORKING-TABLE-SUBSCRIPTS.		
05 TAX-TABLE-SUBSCRIPT	PIC S9(4)	COMP SYNC.
05 DEDUCTION-TABLE-SUBSCRIPT	PIC S9(4)	COMP SYNC.
01 PAYROLL-REPORT-LINE-AREA.		
05 PAYROLL-EMPLOYEE-ID	PIC X(5).	
05 FILLER	PIC X(3)	VALUE SPACES.
05 PAYROLL-EMPLOYEE-NAME	PIC X(20).	
05 FILLER	PIC X(4)	VALUE SPACES.
05 PAYROLL-GROSS-PAY	PIC \$\$\$,\$\$\$,99.	

في الصيغة الثانية ، تم تجميع العناصر الستة الاصلية على مستوى 77 في ثلاثة أجزاء : عدادات ومفاتيح برنامج ودلائل. تعالج كل مجموعة عناصر كمجموعة حقول على المستوى 01 . اسم البيانات على المستوى 01 يكون وصفيا لفئة عناصر البيانات المعرفة . وهذه الصيغة الثانية أفضل كثيرا في تنظيمها عن الصيغة الاولى .

مثال ٥ - ٥٣ :

بالرغم من أن جزء PIC لايسرى على مجموعة عناصر.. فيمكن تحديد SYNCHRONIZED, USAGE على مستوى المجموعة ، وفي هذه الحالة.. تسرى على كل العناصر الموجودة في المجموعة . ونشعر بأن هذه العملية يجب أن تستخدم بشكل محدود، إذا استخدمت على الاطلاق ، حيث إنها تزيد من احتمال عدم تمييز مبرمج الصيانة USAGE الصحيحة للعنصر. الموقع الآمن نسبيا لاستخدام USAGE لمجموعة في تجميعات العناصر المستقلة (عناصر المستوى 77) في قسم مخزن العمل (كما في مثال ٥ - ٥٢ ، WORKING- COUNTERS COMP-3) .

جزء القيمة

عندما يستخدم جزء

$$\left[\underline{\text{VALUE}} \text{ IS } \left\{ \begin{array}{l} \text{literal} \\ \text{figurative constant} \end{array} \right\} \right]$$

فإنه يحدد قيمة عنصر بيانات مخزن العمل عند بدء تنفيذ البرنامج . إذا لم يستخدم جزء VALUE .. فيكون لعناصر البيانات محتويات غير متوقعة في البداية ، ويقول المبرمجون إن مثل هذه العناصر تحتوي على نفايا أو إنها غير معرفة . لاتهم القيم الابتدائية لعدد من العناصر نظرا لأن منطق البرنامج يدخل بيانات ذات معنى فيها (حاذفا النفايا) قبل محاولة استخدام محتويات الموقع .

هناك نوعان من الثوابت، وستة أنواع من الثوابت الاستعارية .

• **الثوابت غير العددية** Nonnumeric literals : يحتوى الثابت غير العددي على أى سلسلة من الرموز موضوعة بين علامتى تنصيص (تسمى quoted string - القسم الثانى - الفصل الثانى) . إذا كانت هناك رغبة في تمثيل علامة تنصيص في ثابت غير عددي.. تستخدم علامتى تنصيص متتاليتين ، ولا تعتبر إلا علامة تنصيص واحدة كجزء من الثابت .

مثال ٥ - ٤٤ :

05 CUSTOMER-NAME PIC X(15) VALUE ""KWIK"" CLEAN".

يحدد القيمة "KWIK" CLEAN (يمثل كل زوج من علامات التنصيص المزدوجة داخل الثابت علامة تنصيص مزدوجة واحدة في القيمة) .

تستخدم بعض المترجمات علامة تنصيص فردية (أو فاصلة) ، بدلا من علامة التنصيص المزدوجة ، ويحدد كويل 1974 النمطى استخدام علامات تنصيص مزدوجة ، إلا إنه يجب على المبرمج أن يرجع - بالطبع - إلى نظام الكمبيوتر المتاح له استخدامه .

05 CUSTOMER-NAME PIC X(15) VALUE 'PERELMAN'S PETS'.

• **الثوابت العددية** Numeric literals : تتكون من الأرقام من صفر إلى ٩ وإشارة الموجب وإشارة السالب والعلامة العشرية . تعتمد أقصى قيمة للثابت العددي على نظام الكمبيوتر وهي ١ - ١٨١٠ في كويل IBM OS/VS (أى ١٨ خانة) . فإذا لم تحدد الإشارة .. تفترض الثوابت العددية أنها موجبة . وإذا لم توجد علامة عشرية في ثابت عددي ، فيقال إنه من النوع الصحيح integer .

• **الثوابت الاستعارية** Figurative constants : الثابت الاستعارى هو كلمة كويل محجوزة، تمثل قيمة ثابتة يمكن استخدامها في أى مكان في البرنامج في موقع الثابت المناظر لها . ويعرض جدول ٥ - ٢ الثوابت الاستعارية الموجودة في كويل IBM SO/VS

القيمة	نوع الثابت الذى تمثله	كلمة كويل المحجوزة
صفر (يتكرر عدد مناسباً من المرات ، يضبط المترجم الصفر على إنه COMP ، أو DISPLAY)	عددي	ZERO ZEROS (or ZEROES)
فراغ (يتكرر عدد مناسب من المرات)	غير عددي	SPACE SPACES
أعلى رمز في تسلسل التتابع (يتكرر عدداً مناسباً من المرات) .	غير عددي	HIGH-VALUE HIGH- VALUES
أقل رمز في تسلسل التتابع (يتكرر عدد مناسب من المرات) .	غير عددي	LOW- VALUE LOW- VALUES
علامة تنصيب (فردية أو مزدوجة) وتكرر عدداً مناسب من المرات .	غير عددي	QUOTE QUOTES
يمكن أن يكون literal أى سلسلة رموز بين علامتى تنصيب (تتكرر السلسلة) .	غير عددي	ALL literal

جدول ٥ - ٢

مثال ٥ - ٥٥ :

لاحظ استخدام الثوابت والثوابت الاستعارية فيما يلى :

WORKING-STORAGE SECTION.

01 ALL-WORKING-STORAGE.

```

05 FILLER                PIC X(30)
                           VALUE "WORKING-STORAGE BEGINS HERE".
05 LAST-RECORD-FLAG      PIC X(3)
                           VALUE HIGH-VALUES.
05 UNDERLINE-AREA        PIC X(100)  VALUE ALL "-".
05 BLANK-LINE-AREA        PIC X(100)  VALUE SPACES.
05 INVALID-RECORD-COUNT  PIC S9(4)    COMP-3  VALUE ZERO.

```

PROCEDURE DIVISION.

```

MOVE SPACES TO OUTPUT-MESSAGE-AREA
MOVE "INVALID PART NUMBER" TO OUTPUT-MESSAGE
MOVE 200.95 TO DAILY-QUOTA
MOVE ALL "" TO PAID-OUT-MESSAGE-AREA
MOVE LOW-VALUES TO EMPLOYEE-MASTER-ID

```

لاحظ بصفة خاصة استخدام الثابت غير العددي في تحديد بداية مخزن العمل ، وهذا يسهل من إيجاد مخزن العمل في الزحام dump .

القيمة والنقل

كما سبق أن رأينا فإن جزء VALUE يمكن استخدامه في وضع قيم ابتدائية في عناصر مخزن العمل .

05 INVALID-RECORD-COUNT PIC S9(4) COMP-3 VALUE ZERO.

ويمكن الان استخدام عبارة MOVE من جزء الإجراءات في بداية تنفيذ البرنامج، لوضع قيم ابتدائية للعناصر كذلك .

MOVE ZERO TO INVALID-RECORD-COUNT

ونقدم المقترحات التالية الخاصة بهاتين الطريقتين :

- (١) إذا احتفظ العنصر الذي تحدد له قيمة ابتدائية بقيمته الابتدائية أثناء تنفيذ البرنامج .. استخدم جزء VALUES .
- (٢) إذا تغيرت قيمة عنصر البيانات أثناء تنفيذ البرنامج، ويتطلب منطق البرنامج أن يكون للعنصر قيمة ابتدائية محددة.. استخدم عبارة MOVE .
- (٣) إذا لم يكن العنصر في مخزن العمل (باستثناء العناصر الموجودة على المستوى 88 ، انظر القسم الثامن - من الفصل السابع) فلا يمكن استخدام جزء VALUE .
- (٤) إذا كانت القيمة الابتدائية لعنصر البيانات غير مناسبة للبرنامج .. فلا تضع قيمة ابتدائية للعنصر .

٥ - ٢٠ نهطيات كتابة الشفرة

نعيد هنا صياغة نهطيات كتابة الشفرة لجزء التعريف، وجزء الأوساط وجزء البيانات .

- (١) مقاطع جزء التعريف مشمولة في كويل ANS النمطي؛ لان المعلومات التي تحملها مهمة (لاتهمل محتويات جزء التعريف).
- (٢) هناك مرونة معتبرة في كتابة جزء التعريف ؛ بما يجعله سهل القراءة .
- (٣) استخدم أسطر تعليقات في نهاية جزء التعريف لتلخيص ما يؤديه البرنامج .
- (٤) استخدم نقاط مرتبة عندما تكون هناك عدة أجزاء (كما هو الحال في مقطع الأسماء الخاصة وفي وصف الملف FD) .
- (٥) استخدم أرقام مستويات متزايدة بأكثر من 01 ، سامحا بإدخال أرقام بينها مستقبلا . (يستخدم في هذا الملخص (01، 05 ، 10 ، 15 ، 20 ، ٠٠٠) .
- (٦) قم بعمل ترحيل لتوضيح هيكل أرقام المستويات .
- (٧) يجب أن تكون لكل اسم بيانات سابقة prefix (أو لاحقة suffix) تصاحبه بالسجل الذي يتبعه العنصر .
- (٨) يجب أن تكون بقية اسم البيانات وصفية بقدر الامكان (تذكر أنك تستطيع استخدام 30 رمز في الاسم الواحد) .
- (٩) اكتب عبارات SELECT مرتبة في احدى الصيغ النمطية (مثل ملفات المدخلات ثم ملفات المدخلات والمخرجات ، ثم ملفات المخرجات ، واستخدام نفس الترتيب في FD) .

(١٠) لاستخدم عناصر على المستوى 77 في مخزن العمل ، وأجمع بدلا من ذلك العناصر في فئات معنوية ، تحت اسم وصفى على المستوى 01 .

(١١) استخدم ترتيبا نمطيا معيناً في كتابة عناصر مخزن العمل ليشتمل على : مفاتيح البرنامج، والإشارات، والعدادات، والثوابت الاستعارية، والثوابت، ومناطق الإجماليات (المركبات) والدلائل والجدول، ونسخ العمل للسجلات المنطقية للملفات . ويجب أن ترتب هذه الأخيرة مثل ترتيب ظهورها في FD ، عرف الملفات الطباعة نسخ العمل لأسطر العناوين أولاً، يليها أسطر الجسم وأخيراً أسطر الإجماليات والنهاية.

(١٢) كلما كان ممكناً اضبط VALUE , SYNC , USAGE , PIC في الأعمدة . بحيث يبدأ كل منها من عمود معين .

(١٣) إذا كان ممكناً .. حدد وظيفة عنصر البيانات في اسم البيانات ، مثل جعل كل مفاتيح البرنامج تنتهي بكلمة SWITCH أو باختصارها SW، وكل الإشارات تنتهي بكلمة FLAG، ومناطق الإجماليات تنتهي بكلمة TOTAL، وتنتهي كل العدادات بكلمة COUNTER أو COUNT .

(١٤) تقدم بعض المترجمات توسعات في كويل ANS ، تسمح للمبرمج بفصل العناصر في قائمة برنامج المصدر، لتسهيل القراءة أكثر . فمثلاً.. يتسبب كويل IBM OS/VS باستخدام EJECT المكتوبة في أى مكان من السطر في المنطقة B ، في بدء السطر التالي من القائمة في قمة صفحة جديدة . وباستخدام SKIP1 أو SKIP2 أو SKIP3 المكتوبة في أى مكان في المنطقة B من السطر ، يترك المترجم العمود المحدد له من الأسطر دون طباعتها في القائمة، ويمكن استخدام هذه السمات: للتأكد أن كل جزء، وكل قسم يبدأ في صفحة جديدة من القائمة ، وأن المقاطع مفصولة عن بعضها البعض (مع وجود أسطر فارغة قبل اسم المقطع) . يمكن تشغيل SKIP 1 ، و SKIP 2 ، و SKIP 3 أسرع بواسطة المترجم ، عما في حالة استخدام أسطر فارغة موجودة داخل البرنامج لعمل الفراغات .

اسئلة مراجعة

- ٥ - ١ ما الغرض من جزء البيانات ؟ ومن قسم الملفات ومن قسم مخزن العمل ؟
- ٥ - ٢ ما نوع عناصر البيانات المتوقع وجوده في مخزن العمل ؟
- ٥ - ٣ ما الغرض من الأجزاء التي تكتب على المستوى 01 في جزء البيانات ؟
- ٥ - ٤ وضع العلاقة بين FD وعبارات SELECT ؟
- ٥ - ٥ ما المعلومات التي تحدد في FD ؟
- ٥ - ٦ وضع كيف يوفر التجميع من المواقع على الشريط أو القرص .
- ٥ - ٧ ميز بين السجلات المنطقية والسجلات الواقعية ؟
- ٥ - ٨ عرف معامل التجميع .
- ٥ - ٩ ما المقصود بـ « فراغات ما بين السجلات، وفراغات ما بين المجموعات » ؟
- ٥ - ١٠ وضع كيف يسرع التجميع من تشغيل الملفات التابعة للقرص أو الشريط .
- ٥ - ١١ وضع تأثير التجميع على تشغيل الملفات عشوائياً .
- ٥ - ١٢ وضع كيف تعالج السجلات متغيرة الطول بواسطة نظم IBM OS/VS .

- ٥ - ١٣ وضع كيف تحسب قيم BLOCK CONTAINS لسجلات متغيرة الطول .
- ٥ - ١٤ لماذا يعد جزء RECORD CONTAINS اختياريا ؟ ومتى يكون اجباريا ؟
- ٥ - ١٥ وضع كيف تحسب قيم RECORD CONTAINS لسجلات متغيرة الطول .
- ٥ - ١٦ ما المعنى الخاص لجزء BLOCK CONTAINS 0 RECORD فى كويل IBM SO/VS ؟
- ٥ - ١٧ وضع استخدام عناوين الملفات للقات الشرائط والأقراص .
- ٥ - ١٨ ما حجم جدول المحتويات VTOC أو الدليل ؟
- ٥ - ١٩ ما الجزء الوحيد اللازم ظهوره فى FD ؟
- ٥ - ٢٠ أى الملفات يفضل أن تكون لها عناوين ملفات ؟ وأيها يفضل ألا تكون لها عناوين ملفات ؟
- ٥ - ٢١ لماذا يستخدم DATA RECORD بصفة متكررة ؟ ومتى يمكن أن يكون مفيدا ؟
- ٥ - ٢٢ اذكر مثلا ملف به أكثر من نوع واحد من السجلات المنطقية . ووضح كيف يمكن استخدام شفرة السجلات record codes فى التمييز بين الأنواع المختلفة .
- ٥ - ٢٣ عرف العناصر التالية المصاحبة لجزء الخطية LINAGE : (أ) صفحة منطقية (ب) جسم الصفحة (ج) منطقة نهاية (د) هامش علوى (هـ) هامش سفلى .
- ٥ - ٢٤ مع أى أنواع الملفات يجب استخدام جزء LINAGE ؟
- ٥ - ٢٥ ما منطقة المدخلات والمخرجات I/O area أو الذاكرة الاحتياطية buffer ؟
- ٥ - ٢٦ وضع العلاقة بين العناصر المعرفة على المستوى 01 فى FD ملف، والذاكرات الاحتياطية للملف .
- ٥ - ٢٧ ما أرقام المستويات المسموح باستخدامها ؟
- ٥ - ٢٨ ما الشئ الخاص برقم المستوى 01 ؟
- ٥ - ٢٩ وضع كيفية استخدام أرقام المستويات فى توضيح هيكل السجل .
- ٥ - ٣٠ ما هى أهمية الترحيل فى جزء البيانات ؟
- ٥ - ٣١ عرف : (أ) مجموعة العناصر (ب) العنصر الفردى .
- ٥ - ٣٢ قدم بعض المقترحات لعمل أسماء لأسماء بيانات .
- ٥ - ٣٣ عندما يكون للملف أكثر من نوع واحد من السجلات المنطقية ، لماذا يكون كل من النوعين متفقاً بصفة عامة فى بايت معين، أو فى مجموعة من البايت ؟
- ٥ - ٣٤ وضع كيف تسرع الذاكرات الاحتياطية المتعددة من تشغيل الملفات التابعة .
- ٥ - ٣٥ فيم يستخدم FILLER ؟
- ٥ - ٣٦ صف معنى التنقيح editing .
- ٥ - ٣٧ ما الغرض من جزء PICTURE ؟
- ٥ - ٣٨ كيف يمكن تكرار رمز صورة بسهولة فى PICTURE ؟
- ٥ - ٣٩ وضع استخدام رمزى الصورة A و B ؟

- ٥ - ٤٠ لماذا يتكرر استخدام X دائما كرمز صورة ؟
- ٥ - ٤١ ناقش مايجب استخدامه في PICTURE لتمثيل أعداد .
- ٥ - ٤٢ وضح استخدام S و 9 و V في تمثيل الأعداد . ميز بين هذه الرموز والرمز X .
- ٥ - ٤٣ عرف : (أ) الحذف (ب) التقريب .
- ٥ - ٤٤ لماذا لايتطلب S أو V موقع تخزين في عنصر البيانات ؟
- ٥ - ٤٥ متى يجب استخدام S ؟ ولماذا ؟ ومتى يجب حذفها ؟
- ٥ - ٤٦ ما نتيجة عنصر عددي له رموز S و 9 و V في صورته ويطبع كما هو في نظام IBM ؟
- ٥ - ٤٧ وضح التتقيح العددي باستخدام Z ، والنقطة والفاصلة و \$ ، و CR ، و DB ، وإشارة سالبة وإشارة موجب .
- ٥ - ٤٨ وضح حماية الشيكات .
- ٥ - ٤٩ وضح الفرق بين التتقيح الثابت fixed والمتحرك floated .
- ٥ - ٥٠ في أى صورة تطبع الأعداد السالبة عادة في تقارير الأعمال ؟
- ٥ - ٥١ وضح استخدام صورة 0 ، و / ، و P .
- ٥ - ٥٢ حدد USAGE المتاح في كويل IBM OS/VS ، وأذكر متى يجب أو لا يجب استخدام كل منها .
- ٥ - ٥٣ متى يجب أن يعرف العنصر بأنه DISPLAY ؟
- ٥ - ٥٤ ماذا يكون COMP و COMP-3 أكثر كفاءة للحقول في الحسابات ؟
- ٥ - ٥٥ اذكر بعض مميزات استخدام COMP أو COMP-3 ، الحقول العددية في سجلات قرص أو شريط ؟
- ٥ - ٥٦ وضح كيف تحدد طول عنصر بيانات كما يلي (أ) DISPLAY (ب) CONP (ج) CONP-3 .
- ٥ - ٥٧ وضح استخدام BLANK WHEN ZERO .
- ٥ - ٥٨ وضح وظيفة JUSTIFIED .
- ٥ - ٥٩ ما وظيفة جزء OCCURS ؟
- ٥ - ٦٠ ناقش البدائل المختلفة لجزء SIGN IS .
- ٥ - ٦١ وضح ما تفعله SYNCHRONIZED .
- ٥ - ٦٢ ما البايث الراكدة slack bytes ؟ اربطها بالسؤال عن متى يجب أو لا يجب استخدام SYNC ؟
- ٥ - ٦٣ وضح استخدام REDEFINES مع إعطاء مثال .
- ٥ - ٦٤ ما البديل لاستخدام مستوى 77 في مخزن العمل ؟
- ٥ - ٦٥ ما معنى نفايا garbage ؟
- ٥ - ٦٦ اذكر قواعد عمل ثوابت الكويل، وأذكر الثوابت الاستعارية .
- ٥ - ٦٧ وضح استخدام جزء VALUE .
- ٥ - ٦٨ متى يجب أن توضع قيمة ابتدائية لعنصر بيانات باستخدام جزء VALUE ؟ ومتى يجب حدوث ذلك بعبارة MOVE من جزء الإجراءات ؟

مسائل محلولة

٥ - ٦٩ ما الخطأ فى محتوى قسم مخزن العمل التالى ؟

05 MAXIMUM-TOTAL-SALES PIC S9(5)V99 VALUE HIGH-VALUES.

HIGH-VALUES و LOW-VALUES هى ثوابت استعارية غير عددية ، ويمكن استخدامها مع PIC X فقط

ولا يمكن استخدامها فى الحسابات .

٥ - ٧٠ بمعرفة مايلى :

SELECT CUSTOMER-ORDER-FILE ASSIGN TO ORDERS.
SELECT ORDER-EDIT-REPORT ASSIGN TO EDITLOG.
SELECT CUSTOMER-MASTER-FILE ASSIGN TO CUSTMAST.

ما أسماء FD التى يجب استخدامها فى قسم الملفات ؟

FILE SECTION.

FD CUSTOMER-ORDER-FILE

FD ORDER-EDIT-REPORT

FD CUSTOMER-MASTER-FILE

٥ - ٧١ افترض أن CUSTOMER- ORDER- FILE فى المسألة السابقة يحتوى على سجلات، طول كل منها 65 بايت ،
ويعامل تجميع 40 لها . وأنها مخزنة حالياً على قرص . اذكر FD الكامل لها .

FD CUSTOMER-ORDER-FILE
BLOCK CONTAINS 40 RECORDS
RECORD CONTAINS 65 CHARACTERS
LABEL RECORDS ARE STANDARD

حيث إن الملف موجود على قرص ، .. فإن المطلوب هو عناوين نمطية للملف . فى نظم IBM ، يكون من الأفضل
استخدام BLOCK CONTAINS 0 RECORDS ؛ بحيث يمكن أخذ معامل التجميع من عنوان الملف الموجود فعلاً .

٥ - ٧٢ إذا كان الملف الموجود فى المسألة السابقة محتوياً على 2000 سجل ، ما العدد اللازم من المدخلات من القرص لتشغيل
الملف تتابعياً ؟

مع تجميع 40 سجل فى المجموعة ، يتطلب 2000 سجل عدد 2000/40 أى 50 مجموعة . لأن كل عملية مدخلات من
القرص تدخل مجموعة .. فلا تكون هناك حاجة إلا إلى 50 عملية مدخلات فقط .

إذا لم يستخدم التجميع مع الملف .. فلا يوجد إلا سجلاً منطقياً واحداً فى كل سجل واقعى . وعلى هذا .. يجب أن يكون
هناك عملية مدخلات لكل سجل منطقى؛ أى 2000 عملية للملف كله .

٥ - ٧٣ لماذا يعتبر عدد عمليات مدخلات أو مخرجات القرص مقياسا مهما لسرعة تشغيل الملف ؟

على عكس الاتصال بالبيانات في ذاكرة الكمبيوتر ، والذي يمكن أن يحدث في نانوثانية (10^{-9} ثانية) أو أقل ، فأحيانا .. يشمل الاتصال بالبيانات من القرص حركة ذراع اتصال (الا إذا كانت للقرص رأس ثابتة لكل مسار أو إلا إذا ما حدث أن تواجدت الذراع المتحركة في المكان المطلوب بالصدفة) ، ويشمل دائما تأخير دوران بينما تدور المجموعة المطلوبة تحت الرأس . وهذه هي عمليات ميكانيكية ، وهي بطيئة جدا ، بالمقارنة بالعمليات الالكترونية للكمبيوتر : لأقرص IBM من طراز 3308 ، يكون متوسط وقت الحركة ١٦ ميللي ثانية ومتوسط وقت تأخر الدوران ٨,٣ ميللي ثانية . إذا ما احتوى تشغيل الملف على عدد كبير من الاتصال الواقعي بالقرص ، فمجموع هذه الاوقات يصبح وقتا كبيرا .

٥ - ٧٤ كيف يؤثر التجميع على استغلال المواقع المتاحة على القرص أو الشريط ؟

يوفر التجميع المواقع على القرص أو الشريط عن طريق حذف بعض فراغه ما بين السجلات . فمثلاً يحتاج الملف الموجود في المسألة رقم ٧٢ الى ٤ مسارات على قرص من طراز 3308 اذا كان معامل التجميع 40 . اما إذا كان معامل التجميع 1 فإنه يحتاج إلى 25 مسارا .

٥ - ٧٥ افرض ان ORDER- EDIT- REPORT في المسألة رقم ٧٠ ، سيكون مخرجات على طابع يطبع ١٢٢ رمزا في السطر الواحد . اكتب FD كاملا لهذا الملف .

```
FD ORDER-EDIT-REPORT
RECORD CONTAINS 132 CHARACTERS
LABEL RECORDS ARE OMITTED
LINAGE IS 60
... WITH FOOTING AT 55
      LINES AT TOP 3
      LINES AT BOTTOM 3
```

جزء BLOCK CONTAINS محذوف ، لتحديد ان ملف الطباعة غير مجمع ، بالنسبة للقات الطباعات يكون هذا الجزء هو LABEL RECORDS ARE OMITTED ، يجعل جزء الخطية LINAGE من الاسهل لعبارات WRITE من جزء الإجراءات أن تتحكم في موقع الأسطر في الصفحة . يمكن إلا تستخدم قنوات التحكم في العربية مع هذا الملف ، حيث إن LINAGE محدد .

٥ - ٧٦ افرض أن CUSTOMER- MASTER- FILE في المسألة رقم ٧٠ ، له نوعان من السجلات المنطقية : REG- PREDEFINED- CUSTOMER- RECORDS ، وطول كل منها 200 بايت بينما ULAR- CUSTOMER- RECORDS طول كل منها 350 بايت . اكتب FD كاملا ، اذا كان معامل التجميع 40 . استخدم BLOCK CONTAINS ... RECORDS .

FD CUSTOMER-MASTER-FILE
BLOCK CONTAINS 40 RECORDS
RECORD CONTAINS 200 TO 350 CHARACTERS
LABEL RECORDS ARE STANDARD
DATA RECORDS ARE
REGULAR-CUSTOMER-RECORD
PREFERRED-CUSTOMER-RECORD

01 REGULAR-CUSTOMER-RECORD ...
01 PREFERRED-CUSTOMER-RECORD ...

٥ - ٧٧ حل المسألة رقم ٧٦ ، مستخدما BLOCK CONTAINS... CHARACTERS .

FD CUSTOMER-MASTER-FILE
BLOCK CONTAINS 8164 TO 14164 CHARACTERS
RECORD CONTAINS 200 TO 350 CHARACTERS
LABEL RECORDS ARE STANDARD
DATA RECORDS ARE
REGULAR-CUSTOMER-RECORD
PREFERRED-CUSTOMER-RECORD

لايشتمل RECORD CONTAINS على بايت مستخدم في كلمات واصف السجل والمجموعة ، إلا أنه عند استخدام جزء CHARACTERS فيجب أن يحتوى BLOCK CONTAINS على منطقة من ٤ بايت . عدد البايت المطلوب هو ٨١٦٤ وذلك لشغل مجموعة من الحجم الصغير : وهو ٤٠ سجل منطقي ، وطول كل منها طوله ٢٠٠ بايت مضافا إليها ٤٠ كلمة واصف سجل ، طول كل منها ٤ بايت ، مضاف إليها كلمة واصف مجموعة من الحجم الكبير هو ١٤١٦٤ بايت .
في كويل IBM OS/VS يفضل تحديد BLOCK CONTAINS CHARACTERS بحيث يمكن اخذ حجم المجموعة من عنوان الملف (حيث إن هذا يتواجد فعلا في الملف الذي له عناوين نمطية) .

٥ - ٧٨ إذا لم يكن الملف موجودا ، ويراد أعددته بواسطة البرنامج ، هل يمكن استخدام BLOCK CONTAINS 0 ؟
نعم ، إلا أنه في هذه الحالة.. يجب أن تحدد عبارة لغة تحكم العمل التي تعرف الملف حجما للمجموعة وإلا فإنه سيحدث خطأ .

٥ - ٧٩ صنف وحدات المدخلات والمخرجات التي سبق مناقشتها في هذا الكتاب ، من ناحية : إذا كانت تدعم أو لا تدعم عناوين ملفات .

سجلات العناوين مخنوفة	سجلات العناوين نمطية
١ - قارئ البطاقات	١ - قرص له رأس متحركة
٢ - مثقب البطاقات	٢ - قرص له رأس ثابتة وأسطوانات
٣ - الطابع	٣ - شريط مغناطيسي (على بكره)
	٤ - شريط مغناطيسي (في كاسيت)
	٥ - قرص مرن

العناوين مطلوبة لكل أنواع ملفات الاقراص . وتستخدم العناوين - بصفة عامة - لكل أنواع ملفات الشرائط : نظرا للأمن والراحة التي تقدمها .

٥ - ٨٠ اكتب عبارات جزء إجراءات ، تشمل : شفرات سجلات ، وتطبع عناوين بريدية من ملف مثالي رقم ٥ - ١٦ ورقم ٥ - ١٧ .

```

READ MAILING-LABEL-FILE RECORD
IF MAILING-LABEL-NAME-ST-CODE IS EQUAL TO "1"
    MOVE MAILING-LABEL-NAME-ST-NAME      TO LABEL-NAME
    MOVE MAILING-LABEL-NAME-ST-STREET    TO LABEL-STREET
ELSE
    IF MAILING-LABEL-NAME-ST-CODE IS EQUAL TO "2"
        MOVE MAILING-LABEL-CITY-STATE-CITY    TO LABEL-CITY
        MOVE MAILING-LABEL-CITY-STATE-STATE    TO LABEL-STATE
        MOVE MAILING-LABEL-CITY-STATE-ZIP      TO LABEL-ZIP
    ELSE
        DISPLAY "ERROR -- INVALID RECORD CODE: IGNORED"

```

تدخل عبارة READ سجلا منطقيا، أولا من الملف MAILING-LABEL-FILE . تذكر من مثال ١٧ - الفصل الخامس، أن أول بايت من السجل المنطقي يشغلها شفرة سجل تعرف نوع السجل . افترض أن سجلات الاسم والشارع لها ١ في أول بايت، وسجلات المدينة، والولاية، والرمز البريدي لها ٢ في أول خانة . حيث أن أول بايت من كل نوع من أنواع السجلات موصوف بصورة X فلايهم شئ بالنسبة إلى اسم البيانات الذي يستخدمه المبرمج في الإشارة إليه (يصل كل من MAILING-LABEL-NAME-ST-CODE و MAILING-LABEL-CITY-STATE-CODE إلى أول بايت من السجل المنطقي ويفسر على أنه الصورة X) .

بعد عبارة READ ، يستخدم البرنامج عبارة IF في التأكد مما إذا كان أول بايت في السجل المنطقي به 1 أم لا . فإذا كان فيه 1 ، فينقل الاسم والشارع إلى منطقة العنوان ، إما إذا لم يكن هذا هو الحال، فيتأكد مما إذا كان فيه ٢ أم لا . فإذا لم يكن فيه ١ أو ٢ فيطبع البرنامج رسالة خطأ، ويهمل السجل المنطقي . لاحظ كيفية استخدام أسماء البيانات في عبارات MOVE وأنها تعتمد تماما على شفرة السجل

٥ - ٨١ يمكن استخدام كل من PIC A و PIC X مع بيانات حرفية . أيهما أفضل ؟

يستخدم عديد من مبرمجي الكويل PIC X بدلا من PIC A في وصف البيانات الحرفية . لا يوجد عيب في عمل ذلك ، إلا أنه هناك شئ من المميزات فعلا : فبعض عبارات جزء الإجراءات تسرى على PIC X ، ولا تسرى على PIC A . (انظر الفصلين السادس والسابع) .

٥ - ٨٢ صنف عناصر البيانات التالية صورها على أنها (i) حرفية أو (ii) حرفية عدية أو (iii) عدية أو (iv) عدية للتنقيح .

- | | | |
|--------------|----------------------|-------------------------|
| (a) AABAAA | (e) 99/99/99 | (i) A(10) |
| (b) XXBXXX | (f) \$\$\$,\$\$\$,99 | (j) 9(3)V99 |
| (c) S9(3)V99 | (g) S99PP | (k) \$\$\$,\$\$\$,99BCR |
| (d) S9(3).99 | (h) X(7) | (l) Z999 |

- الحرفية : (a) و (i)
- الحرفية عددية : (b) و (h)
- العددية : (c) و (g) و (j)
- العددية للتنقيح : (d) و (e) و (f) و (k) و (l)
- ٥ - ٨٣ اكتب صورة مناسبة PICTURE ، واستخدم USAGE إذا لم يكن DISPLAY ، لكل من عناصر البيانات التالية (إذا كان هناك أكثر من خيار واحد ممكن، اكتب أفضلها أولاً) .
- (a) رقم الضمان الاجتماعى (بدون شرطة)
- (b) مبلغ فى شيك لزيادة قيمته عن ١٠٠٠ دولار .
- (c) رقم الضمان الاجتماعى (بشرط) .
- (d) الاسم الأخير (يمكن ان يصل الى ٢٠ رمز) .
- (e) عدد ساعات العمل وقت إضافى (مقربا لأقرب خانة عشرية واحدة) - ملف شريط .
- (f) عدد الخوابير فى المخزن (يمكن ان يصل الى 99,999) - ملف قرص .
- (g) موازنة قرض (حتى ٥٠٠,٠٠٠,٠٠٠) - ملف بطاقات .
- (h) موازنة قرض (حتى ٥٠٠,٠٠٠,٠٠٠) - ملف قرص .
- (i) موازنة قرض (حتى ٥٠٠,٠٠٠,٠٠٠) - تقرير مطبوع .
- (j) ملف موازنة قرض (حتى ٥٠٠,٠٠٠,٠٠٠) - تقرير مطبوع .
- (k) عدد ساعات العمل وقت اضافى (لأقرب ٠.١) - ملف بطاقات .
- (l) عدد ساعات العمل وقت إضافى (لأقرب ٠.١) - تقرير مطبوع .
- (m) عدد الخوابير فى المخزن (حتى 99,999) - تقرير مطبوع .
- (n) تكلفة الشحن للرجل (لأقرب ٠.١ سنت) - ملف بطاقات .
- (o) تكلفة الشحن للرجل (لأقرب ٠.١ سنت) - ملف قرص .
- (p) تكلفة الشحن للرجل (لأقرب ٠.١ سنت) - تقرير مطبوع .
- (q) تاريخ فى الصورة yymmdd (بدون شرط) - ملف بطاقات .
- (r) تاريخ فى الصورة yymmdd (بدون شرط) - ملف قرص .
- (s) تاريخ فى الصورة yymmdd ، منقح للطباعة .
- (t) شفرة سجل من خانة واحدة - ملف بطاقات .
- (u) شفرة سجل من خانة واحدة - ملف قرص .
- (v) شفرة سجل من خانة واحدة - تقرير مطبوع .
- (w) تعليمات شحن (حتى ٧٠ رمز) - ملف بطاقات ، و ملف قرص ، وتقرير طباعة .
- (x) رقم جزء (سبعة أرقام) - ملف بطاقات .

(y) رقم جزء (سبعة أرقام) - ملف قرص .

(z) الراتب الذي ترغب في الحصول عليه (ويمكن التعبير عنه بكيول IBM OS/VS) - تقرير مطبوع .

٥ - ٨٤ اذكر الطول ، بعدد البايت ، لكل عنصر بيانات معروف في المسألة السابقة (افترض استخدام نظام IBM OS/VS ، اذكر أطوال كل من COMP و COMP-3 عندما يكون ذلك مناسباً) .

(a) ٩ (مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(b) ٩ (مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(c) ١١ (مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(d) ٢٠ (مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(e) COMP-3 : 2 [PIC لها ٢ أرقام ولهذا تأخذ $2/3$] $1 + 2 = 3$ بايت] .

COMP : ٢ (يمكنها إمساك من ١ إلى ٤ أرقام) .

(f) COMP-3 : ٣ [PIC لها ٥ أرقام ولهذا تأخذ $5/5$] $1 + 3 = 4$ بايت] .

COMP : ٤ (يمكنها إمساك من ٥ إلى ٩ أرقام) .

(g) ٨ (مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(h) COMP-3 : 5 [PIC لها ٨ أرقام ولهذا تأخذ $8/8$] $1 + 5 = 6$ بايت] .

COMP : ٤ (يمكنها إمساك من ٥ إلى ٩ أرقام) .

(i) ١٢

(j) ٣ (لا يحتاج S أو V أى موقع إضافياً) .

(k) ٤ (النقطة تمثل علامة عشرية واقعية وتشغل بايت واحداً) .

(l) ٥

(m) ٧ (مع إشارة سالبة متخلفة ، أو ٦ بدونها) .

(n) ٦ (لا يحتاج S أو V أى موقع إضافياً) .

(o) COMP-3 : ٤ [PIC لها ٦ أرقام ولهذا تأخذ $6/6$] $1 + 4 = 5$ بايت] .

COMP : ٢ (يمكنها إمساك من ٥ إلى ٩ أرقام) .

(p) ٧

(q) ٦

(r) ١

(s) ١

(t) ٧

(u) ٦

(v) ٨

١ (w)

٧٠ (x)

٧ (y)

١٧ (z) (يحتاج كل رمز صورة في صورة تنقيح إلى بايت واحد) .

٥ - ٨٥ بمعرفة العبارة DATA - ITEM MOVE ZEROS TO SAMPLE- DATA ... بين النتائج عندما يكون SAMPLE- DATA- ITEM معرفاً كما يلي :

- | | | |
|-------------------------|---------------------|------------------|
| (a) PIC S9(3)V99 COMP-3 | (b) PIC S9(3)V99 | (c) PIC ZZ.ZZ |
| (d) PIC \$**,***.** | (e) PIC \$**,***.** | (f) PIC Z,ZZZ.99 |
| | BLANK WHEN ZERO | |
| (g) PIC Z,ZZZ.Z9 | (h) PIC Z,ZZZ.99 | (i) PIC X(4) |
| | BLANK WHEN ZERO | |
| (j) PIC A(4) | (k) PIC XX/XX/XX | (l) PIC XXBXXBXX |

(a) ٠٠٠ + (صورة COMP-3)

(b) ٠٠٠ + (صورة DISPLAY)

(c) bbbbbb

(d) (\$*****.** تطبع النقطة دائماً مع حماية الشيكات) .

(e) غير صحيح : لا تستخدم BLANK WHEN ZEROS مع حماية الشيكات .

(f) bbbbbb.00

(g) bbbbbb0

(h) bbbbbb

(i) 0000 (أصفار في تمثيل EBCDIC)

(j) غير صحيح : يحتوى PIC A على حروف وفراغات فقط .

(k) 00/00/00

(l) 00b00b00

٥ - ٨٦ بمعرفة العبارة DATA - ITEM MOVE SPACES TO SAMPLE- DATA .. بين النتائج عندما يكون SAMPLE- DATA- ITEM معرفاً كما يلي :

- | | | | |
|----------------|--------------|--------------|------------------|
| (a) PIC X(3) | (b) PIC A(4) | (c) PIC 9(3) | (d) PIC S9(4)V99 |
| (e) PIC XXX/XX | | | |

bbb (a)

bbbb (b)

(c) غير صحيح

(d) غير صحيح

bbb/b (e)

(تذكر ان PIC9 لا يمكن أن تشتمل إلا على أرقام) .

هـ - ٨٧ بمعرفة العبارة التالية : MOVE "ABCDE" TO SAMPLE- DATA - ITEM .. بين النتائج عندما يكون
SAMPLE- DATA - ITEM معرفاً كما يلي :

- (a) PIC X(3) (b) PIC X(3) JUSTIFIED RIGHT (c) PIC XX/XX
(d) PIC X(8) (e) PIC X(8) JUSTIFIED RIGHT

ABC - a (الحذف من ناحية اليمين) .

CDE - b (الحذف من ناحية اليسار لوجود JUSTIFIED RIGHT) .

AB/CD - c (الحذف من ناحية اليمين مع إدخال /) .

ABCDEbbb - d (فراغات من ناحية اليمين) .

e - bbbABCDE (يتسبب جزء JUSTIFIED RIGHT في نقل القيمة من يمين الحقل، وبالتالي تصبح الفراغات
في يسار الحقل).

هـ - ٨٨ ما أهمية SYNC مع عناصر COMP ؟ تعتمد أهمية استخدام SYNC ، مع عناصر COMP على معمارية نظام
الكمبيوتر المستخدم وعلى التنفيذ الفعلي لعناصر COMP . في نظم كمبيوتر IBM-370 . لا يبدو الرمز في وقت التنفيذ
معنوياً . بسبب إدخال بايت راكدة (القسم ١٦ من هذا الفصل) . يجب ألا يستخدم SYNC مع عناصر COMP
الموجودة في سجلات منطقية بصفة عامة) .

هـ - ٨٩ ما نتيجة العبارة MOVE ZEROS TO - A- GROUP- ITEM ، حيث ان العنصر معرف كما يلي :

- 01 A-GROUP-ITEM.
05 A PIC X(2).
05 B PIC S9(3)V99.
05 C PIC S9(5)V99 COMP-3.
05 D PIC S9(4) COMP.

عند معالجة مجموعات عناصر في جزء الاجراءات فدائماً ما تعامل كحقل كبير له صورة من نوع X . وحيث ان طول A-
GROUP- ITEM هو ١٣ بايت (٢ + ٤ + ٥ + ٢) .. فتعامل عبارة MOVE العنصر A- GROUP- ITEM ، كما لو كان
معرفاً بالصورة PIC X (13) . وعلى هذا .. ينقل ١٣ صفراً من نوع DISPLAY إلى بايت العنصر ، التي تدخل أصفار

صحيحة في A و B ، حيث إنهما معرفان بأنهما DISPLAY (تقليديا) . إلا أن C يحتاج صفر من نوع COMP وليست اصفار من نوع DISPLAY . النتيجة هي أن مواقع C و D لا تحتوي على بيانات صحيحة ، وفي الواقع ، معالجة الحقول ، C و D ينتج عنها نتائج غير صحيحة ويمكن أن يفصل تنفيذ البرنامج بنهاية غير طبيعية .

٥ - ٩٠ بمعرفة مايلي :

```
01 SAMPLE-ITEM.
05 A          PIC 9(4)V99.
05 B REDEFINES A.
10 C          PIC 9(2)V9.
10 D          PIC 9(3).
```

ما محتويات C , D بعد تنفيذ العبارة التالية :

MOVE 12,34 TO A

A: PIC 9(4)V99					
0	0	1	2	3	4
C: PIC 9(2)V9			D: PIC 9(3)		
B: PIC X(6)					

شكل (٥ - ١٠)

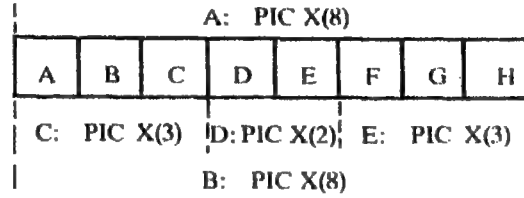
بسبب MOVE يحتوى A على 0012,34 ، ولكن بسبب REDEFINES تشير C , D إلى نفس مواقع ذاكرة الكمبيوتر تماما مثل A . وهذا الموقف موضح في شكل ٥ - ١٠ ويمكن رؤية أن C يحتوى النصف الأيسر من A ، وهو (من نقطة تعريف العنصر C) 00.1 ، ويحتوى D على ٢٣٤ (النصف الأيمن من A ، وهو مجموعة عناصر ، يحتوى على 001234 ، ويمكن أن يعامل على الصورة DISPLAY PIC X (6) .

٥ - ٩١ بمعرفة مايلي :

```
01 SAMPLE-ITEM.
05 A          PIC X(8)  VALUE "ABCDEFGH".
05 B REDEFINES A.
10 C          PIC X(3).
10 D          PIC X(2).
10 E          PIC X(3).
```

حدد ماهية C : في D ؟ في E ؟ في B ؟

حيث إن B يعيد تعريف A .. فإن الموقف يكون موضحا في شكل (٥ - ١١) .



شكل (٥ - ١١)

نظراً لأن B مجموعة عناصر .. فهو يكافئ PIC X (18) . لاحظ أن في إعادة تعريف عنصر .. يجب أن تتساوى أطوال العنصرين . C يحتوى على "ABC" و E يحتوى على "FGH" . و B يحتوى على "ABCDEFGH" حيث إنه يشير إلى نفس مواقع الذاكرة تماماً مثل العنصر A .

٥ - ٩٢ عرف حقلاً يكون جزءاً من سطر مطبوع . يحتوى الحقل على (١) مبلغ من المال (بحد أقصى 50,00.00) المرتبط في تخزين عنصر معين، أو (٢) رسالة "STOCK-OUT" إذا لم يوجد هذا العنصر بالشركة .

```

05 VALUE-OF-INVENTORY      PIC ZZ,ZZZ.99.
05 OUT-OF-STOCK-MESSAGE-AREA REDEFINES VALUE-OF-INVENTORY
                                PIC X(9).

```

هذه المسألة توضح الحاجة إلى ، وسيلة الحصول على ، كل من صورة عددية وصورة غير عددية لنفس الحقل من السجل .
تتسبب العبارة MOVE "STOCK- OUT" TO VALUE- OF- INVENTORY في حدوث خطأ تكويني ، حيث أنه يمكن نقل بيانات عددية فقط الى حقل عددي منقح. أما العبارة: MOVE "STOCK- OUT" TO OUT- OF- STOCK- MESSAGE- AREA فهي صحيحة وتحقق ما هو مطلوب بالضبط .

٥ - ٩٣ ما الخطأ فيما يلي :

```

01 SAMPLE-ITEM.
  05 A          PIC X(5).
  05 B          REDEFINES A.
    10 C        PIC S9(5)    COMP-3.
    10 D        PIC S9(4)    COMP.

```

لا يوجد شيء خطأ . طول B هو ٢ + ٣ + ٥ بايت ، والذي يتفق مع طول A ، كما هو مطلوب . والمبرمج حر في تغيير تخطيط الحقول الجزئية كلية بالنسبة الى PIC ، USAGE وغيرها عندما تستخدم REDEFINES . الشروط الوحيدة هي :
(١) يجب أن يتساوى طول العنصرين .

- (٢) لا يمكن استخدام REDEFINES مع عناصر المستوى 01 في قسم الملفات .
 (٣) العنصر المعاد تعريفه والعنصر الذي يعيد التعريف... يجب أن يكون لهما نفس المستوى .
 ٥ - ٩٤ ما الخطأ فيما يلي :

```
01 SAMPLE-ITEM    PIC X(5).
01 OTHER-ITEM     REDEFINES SAMPLE-ITEM.
    05 A          PIC X(4).
    05 B          PIC X(3).
    04 C          PIC X(2) REDEFINES B.
```

- (١) استخدمت REDEFINES مع عناصر على المستوى 01 ، وهذا صحيح في مخزن العمل إلا أنه خطأ في قسم الملفات .
 (٢) يختلف طول OTHER- ITEM عن طول SAMPLE- ITEM .
 (٣) يختلف طول C عن طول B .
 (٤) بالنسبة للعنصر C يجب أن يكون REDEFINES هو أول جزء في وصف العنصر .
 (٥) رقم مستوى إعادة تعريف C ليس نفس رقم مستوى العنصر المعاد تعريفه B.
 ٥ - ٩٥ ما الخطأ فيما يلي :

WORKING-STORAGE SECTION.

01 PROGRAM-COUNTERS.

05 NUMBER-UNPAID-INVOICES PIC S9(5) COMP-3.

PROCEDURE DIVISION.

ADD 1 TO NUMBER-UNPAID-INVOICES

تأخذ عبارة ADD المحتويات الحالية للعنصر NUMBER- UNPAID- INVOICES وتضيف ١ ، وتعيد النتيجة في نفس العنصر . لكن ما محتويات NUMBER- UNPAID- INVOICES الحالية عندما يبدأ تنفيذ البرنامج ؟ حيث إن المبرمج لم يضع قيمة ابتدائية في عنصر البيانات هذا .. فإن محتوياته ستكون نفايا أو غير معرفة عند تنفيذ ADD ، وهذا خطأ شائع جدا بالنسبة للمبرمجين المبتدئين ، وما يتبع ذلك هو عد إجمالي غير صحيح أو ABEND (نهاية غير طبيعية للبرنامج abnormal end) .

٥ - ٩٦ ماذا يمكن عمله لتصحيح الخطأ في المسألة السابقة ؟

يجب أن نضع قيمة ابتدائية لعنصر البيانات وتكون مساوية صفرا . وحيث ان قيمته تتغير أثناء تنفيذ البرنامج .. فإن الطريقة المفضلة لوضع قيمة ابتدائية، هي باستخدام عبارة MOVE من جزء الإجراءات على النحو التالي :

PROCEDURE DIVISION.
 MOVE ZERO TO NUMBER-UNPAID-INVOICES

 ADD 1 TO NUMBER-UNPAID-INVOICES

حيث إن العنصر NUMBER- UNPAID- INVOICES معرف في مخزن العمل .. فهناك طريقة بديلة ، وهى استخدام جزء VALUE :

05 NUMBER-UNPAID-INVOICES PIC S9(5) COMP-3 VALUE ZERO.

وهذا يؤكد أن NUMBER- UNPAID- INVOICES يحتوى على صفر (بدلاً من نفايا) فى بداية تنفيذ البرنامج .
 (تحجز بعض عمليات الكتابة جزء VALUE للعناصر التى لا تتغير قيمها أثناء تنفيذ البرنامج) .

٥ - ٩٧ لا يفضل عديد من المبرمجين كتابة أوصاف تفصيلية للسجلات المنطقية فى قسم الملفات . بدلا من ذلك فإنهم يقومون بتعريف منطقة مخزن عمل يكون لها نفس الحجم مثل السجل المنطقى . وبمجرد ادخال سجل منطقى من ملف (باستخدام عبارة READ) .. ينقل البرنامج نسخة من السجل المنطقى إلى داخل منطقة مخزن العمل . وتؤدى كل معالجات البيانات بعد ذلك باستخدام نسخة مخزن العمل . وبالمثل ، توضع سجلات المخرجات مع بعضها فى منطقة مخزن عمل لها حجم مناسب ووصف مناسب ، وينقل بعد ذلك إلى منطقة السجل المنطقى من الملف (المستوى 01 فى FD) قبل أن تنفذ عبارة WRITE .

خذ محتويات قسم الملفات التالية وعدلها بحيث يمكن أداء تشغيل السجل فى مخزن العمل .

FILE SECTION.

FD TIME-CARD-INPUT-FILE
 BLOCK CONTAINS 0 CHARACTERS
 RECORD CONTAINS 21 CHARACTERS
 LABEL RECORDS ARE STANDARD

01	TIME-RECORD.	
05	TIME-EMPLOYEE-ID	PIC X(5).
05	TIME-REGULAR-HOURS	PIC 99V9.
05	TIME-OVERTIME-HOURS	PIC 99V9.
05	TIME-EXPENSES	PIC S9(4)V99.
05	TIME-SICK-LEAVE	PIC 99.
05	TIME-VACATION	PIC 99.

FILE SECTION.

FD TIME-CARD-INPUT-FILE
 BLOCK CONTAINS 0 CHARACTERS
 RECORD CONTAINS 21 CHARACTERS
 LABEL RECORDS ARE STANDARD

01 TIME-RECORD PIC X(21).

WORKING-STORAGE SECTION.

01 WORKING-TIME-RECORD-AREA.
 05 WORKING-EMPLOYEE-ID PIC X(5).
 05 WORKING-REGULAR-HOURS PIC 99V9.
 05 WORKING-OVERTIME-HOURS PIC 99V9.
 05 WORKING-EXPENSES PIC \$9(4)V99.
 05 WORKING-SICK-LEAVE PIC 99.
 05 WORKING-VACATION PIC 99.

لاحظ إن وصف السجل في قسم الملفات أصبح تافها - (12) PIC X فقط ، حيث إن طول السجل هو ٢١ . وقد تم نقل الوصف التفصيلي لكل حقل من حقول السجل الى قسم مخزن العمل . لاحظ أن عناصر البيانات REGULAR-HOURS و OVERTIME-HOURS و SICK-LEAVE لم يحدد لها إشارات عمليات . ولا يؤثر هذا على كفاءة برنامج الهدف ، طالما أن هذه العناصر لن تستقبل نتائج حسابات أبدا وأنها لن تكون سالبة على الإطلاق (الشئ الذي يكون مستحيلا ، أن يكون العنصر سالبا ، بدون S) . وقد يختار المبرمج المحافظ أن يستخدم S بالنسبة لهذه العناصر بفرض الأمان .

٥ - ٩٨ يستخدم منهج مخزن العمل الذي سبق مناقشته في المسألة السابقة دائما تقريبا في طباعة ملفات المخرجات ، حيث أن سجلاتها تحتوي على معلومات ثابتة كثيرة ، مثل عناوين الصفحات ، وعناوين الأعمدة وغيرها . يمكن استخدام جزء VALUE في مخزن العمل لأعداد مثل هذه السجلات .

راجع المحتويات التالية بحيث تعد السجلات في مخزن العمل . أنتج كذلك سجلا لطباعة عناوين أعمدة فوق المعلومات المبينة .

FILE SECTION.

FD PAYROLL-REGISTER-FILE
 RECORD CONTAINS 132 CHARACTERS
 LABEL RECORDS ARE OMITTED

01 PAYROLL-REGISTER-LINE.
 05 REGISTER-EMPLOYEE-ID PIC X(5).
 05 FILLER PIC X(10).
 05 REGISTER-REGULAR-HOURS PIC ZZ.9.
 05 FILLER PIC X(10).
 05 REGISTER-OVERTIME-HOURS PIC ZZ.9.
 05 FILLER PIC X(99).

FILE SECTION.

FD PAYROLL-REGISTER-FILE
 RECORD CONTAINS 132 CHARACTERS
 LABEL RECORDS ARE OMITTED

01	PAYROLL-REGISTER-LINE	PIC X(132).
WORKING-STORAGE SECTION.		
01	WORKING-REGISTER-LINE.	
05	WORKING-LINE-EMPLOYEE-ID	PIC X(5).
05	FILLER	PIC X(10) VALUE SPACES.
05	WORKING-LINE-REGULAR-HOURS	PIC ZZ.9.
05	FILLER	PIC X(10) VALUE SPACES.
05	WORKING-LINE-OVERTIME-HOURS	PIC ZZ.9.
05	FILLER	PIC X(99) VALUE SPACES.
01	WORKING-HEADING-LINE.	
05	FILLER	PIC X(15) VALUE "EMPLOYEE ID".
05	FILLER	PIC X(14) VALUE "REGULAR HOURS".
05	FILLER	PIC X(103) VALUE "OVERTIME HOURS".

لاحظ ان PAYROLL- REGISTER- LINE معرف الآن ببساطة بواسطة PIC X (132) ، مع عمل كل الاوصاف الفعلية في WORKING- REGISTER- LINE . يمكن أن توضع فراغات في مناطق FILLER ، لفصل البيانات المطبوعة في السطر ، وذلك باستخدام جزء VALUE (والذي لا يمكن استخدامه عندما يحدث الوصف في قسم الملفات) . لا توجد في WORKING- HEADING- LINE الا محتويات FILLER ولها اجزاء VALUE وذلك لانتاج عناوين الاعمدة المناسبة . اقتنع نفسك ان عنوان كل عمود يبدأ فوق اول رمز معلومات للعمود . (لاحظ ، على أية حال ، أنها ليست متمركزة) .

نماذج بوضحة

من ٥ - ٩٩ إلى ٥ - ١٠٨ عدل التمارين من رقم ٢ - ٢٩ إلى رقم ٢ - ٣٨ : لتعكس معلوماتك الموسعة عن جزء البيانات . غير كل القيم الدولارية إلى دولارات وسننات ، مع إجراء أى تعديلات على تخطيطات السجلات إذا كانت هناك حاجة لذلك . نفع كل المخرجات المطبوعة ، وأجر الحسابات بالكفاءة الممكنة .

الفصل السادس

جزء الإجراءات

Procedure Division

٦ - ١ مقدمة : نهطيات كتابة الشفرة

تخدم أجزاء التعريف والأوساط والبيانات كنوع من المقدمة لجزء الإجراءات ، والذي يحدد فيه المبرمج التعليمات التي يؤديها الكمبيوتر . ويفترض في هذا الفصل أن هذه التعليمات تعمل مع مدخلات وتشغيل ومخرجات الملفات المرتبة تتابعياً (التي يتم الاتصال بها تتابعياً) .

وهيكلياً.. يتكون جزء الإجراءات من أقسام SECTIONS ومقاطع . ويكتب عنوان الجزء -PROCEDURE DIVISION" في المنطقة A ، كما تبدأ كتابة عناوين الأقسام والمقاطع في المنطقة A كذلك . وتتكون الأقسام والمقاطع من جمل sentences والتي تتحدد بنقاط . يمكن أن تتكون كل جملة من عبارة statement واحدة أو أكثر ، والعبارة هي إحدى التعليمات الصحيحة من جزء الإجراءات، التي تبدأ بفعل verb من أفعال الكوبل .

نهطيات كتابة جزء الإجراءات

- (١) اكتب عبارة واحدة فقط في السطر .
- (٢) يجب أن تخصص سطور لعناوين المقاطع والأقسام .
- (٣) يجب أن يكون لكل مقطع وكل قسم غرض معرف تعريفياً جيداً ، ويجب أن يصف الاسم هذا الغرض .
- (٤) رتب عناوين المقاطع والأقسام بإضافة سابقات أو لاحقات للأسماء .
- (٥) أعمل ترحيل (عدد معين من الخانات بدءاً من العمود الثاني عشر) : لتوضيح استمرار العبارة في أكثر من سطر واحد، وتوضيح العلاقات الموجودة بين العبارات .
- (٦) استخدم أسطر فارغة ، أسطر تعليق فارغة ، أو استخدم EJECT , SKIP1 , SKIP2 , SKIP3 (وهي أفضل إذا كانت متاحة) في ترك فراغ بين المقاطع والأقسام وتحديد العناوين .
- (٧) استخدم النقاط عندما تكون هناك فقط ضرورة لها ، واجعلها نقطة مرتبة .

- (٨) استخدم فواصل عندما تكون هناك فقط ضرورة لها .
 (٩) تجنب استخدام أقسام الا إذا كان هذا مطلوباً .
 (١٠) اجعل كل شئ بسيطاً ومباشراً بقدر الإمكان .

٦ - ٢ مدخلات / مخرجات : عبارة الفتح

يجب أن يفتح OPEN كل ملف بنجاح قبل استخدام أى عبارة من عبارات جزء الإجراءات فى تشغيل الملف :

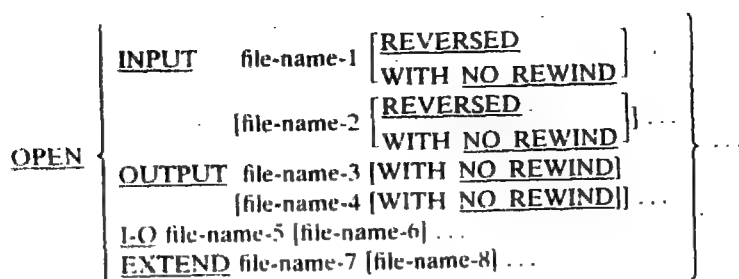
- (١) تحصل عبارة الفتح على ذاكرات احتياطية للملف من مواقع الذاكرة الرئيسية للملف .
 (٢) وتقوم بتشغيل عناوين الملف إذا كانت موجودة (إذا كانت LABEL RECORDS ARE STANDARD محددة فى FD للملف) .

• **ملف مدخلات، أو ملف مدخلات ومخرجات :** تختبر سجلات العناوين الموجودة؛ للتأكد من أن كل المعلومات متاحة، ان الملف يمكن الاتصال به، وأنه معرف تعريفاً صحيحاً.

• **ملف مخرجات :** يتم إنتاج سجلات عناوين للملف .

(٣) ويعد الملف للاتصال بسجلاته المنطقية - عادة تجعله عند بدايته (أى عند بداية أول سجل منطقى) .

التكوين العام لعبارة الفتح OPEN فى شكل (٦ - ١) . كما تحدد الأقواس.. يجب أن تحدد إحدى حالات OPEN الأربع على الأقل ، بعدد الملفات المراد فتحها فى كل حالة .



شكل (٦ - ١)

• **حالة الإدخال INPUT :** يفتح الملف للإدخال فقط ، ويجب أن يكون الملف موجوداً فعلاً ويمكن الاتصال به ، وإلا فيفصل البرنامج فصلاً غير طبعى . ويستخدم فعل القراءة READ ، وهو الفعل الوحيد المستخدم ، فى إدخال سجل منطقى.

• **حالة الإخراج OUTPUT :** يفتح الملف للإخراج فقط ، والفعل الوحيد المستخدم هو فعل الكتابة WRITE . إذا كان الملف موجوداً فعلاً ، فتحذف محتوياته القديمة وتنتج محتوياته الجديدة باستخدام أمر الكتابة فى إخراج السجلات المنطقية . فإذا لم يكن الملف موجوداً ، فيتم إنتاج الملف باستخدام WRITE فى إخراج السجلات المنطقية.

• **حالة الإدخال والإخراج I - O :** يفتح الملف لكل من الإدخال والإخراج ، ويجب أن يكون الملف موجوداً ويمكن

الاتصال به وإلا انتهى البرنامج نهاية غير طبيعية . والأفعال المستخدمة هنا هي REWRITE , WRITE , READ بالنسبة للملف على وحدة اتصال مباشر ، ويمكن استخدام REWRITE في كتابة صيغة معدلة من سجل منطقي في نفس موقع السجل الاصلى من الملف (التجديد في نفس الموقع) .

• حالة موسعة EXTEND : يفتح الملف للمخرجات فقط ، مع تحديد وضعه في نهاية الملف (بعد آخر سجل منطقي) . وعلى هذا ، عندما تستخدم عبارة WRITE في إخراج سجلات منطقية ، فإنها تضيف إلى النهاية الواقعية للملف بعد السجلات الموجودة فعلاً في الملف .

ونادراً ما يستخدم الخياران REVERSED و WITH NO REWIND ، وهما يستخدمان مع الملفات الموجودة على شريط مغناطيسى فقط .

• خيار العكس REVERSED : يمكن استخدام العكس مع ملفات الشرائط المفتوحة كمدخلات فقط ، ويحدد هذا الجزء أن الملف يفتح عند آخر سجل منطقي مع القراءة العكسية ، أى من الخلف إلى الأمام (ولاتوجد هذه الإمكانية في كل مشغلات الشرائط) .

• خيار WITH NO REWIND : يمكن استخدام هذا الخيار (دون إعادة الملف) مع ملفات الشرائط المفتوحة كمدخلات فقط أو كمخرجات فقط . وهو خيار يحدد أن الملف يجب ألا يتغير وضعه أثناء عملية فتحه . وعلى هذا يظل الملف في نفس موقعه عندما تنفذ عبارة الفتح .

مثال ٦ - ١ :

يجب أن تتأطر كل أسماء الملفات المستخدمة في عبارة الفتح التالية الأسماء المستخدمة في عبارة SELECT ، وفي FD .

OPEN	INPUT	EDITED-TIME-FILE
		EMPLOYEE-MASTER-FILE-OLD
	OUTPUT	PAYROLL-REGISTER-REPORT
		EMPLOYEE-MASTER-FILE-NEW
	EXTEND	TIME-RECORD-SUMMARY-FILE

لاحظ نمطية الكتابة هذه لعبارة الفتح : كتبت كل أسماء الملفات في نفس العمود ، وكتبت كل حالات OPEN في نفس العمود ، وذلك مع استخدام الترحيل المناسب .

وكبديل .. يمكن أن يكون لكل ملف عبارة OPEN خاصة به . وبصفة عامة .. فإن فتح عديد من الملفات بعبارة فتح واحدة يأخذ ذاكرة كمبيوتر أكبر لتشغيل OPEN ، ولكنه - في نفس الوقت - يسمح باتمام الفتح بشكل أسرع كثيراً .

مثال ٦ - ٢ :

مايلى يوضح خطأ شائعاً يقع فيه المبرمجون المبتدون .

IDENTIFICATION DIVISION.
PROGRAM-ID. MISTAKE.

ENVIRONMENT DIVISION
INPUT-OUTPUT SECTION.
FILE-CONTROL.

SELECT TIME-CARD-FILE
ASSIGN TO WKLYTIME

SELECT PAYROLL-EDIT-REPORT
ASSIGN TO TIMEEDIT

DATA DIVISION.

FILE SECTION.

FD TIME-CARD-FILE

LABEL RECORDS ARE OMITTED

01 TIME-CARD-RECORD.

05 TIME-CARD-ID PIC X(6).

.....

FD PAYROLL-EDIT-REPORT

LABEL RECORDS ARE OMITTED

RECORD CONTAINS 132 CHARACTERS

.

01 EDIT-LINE.

05 EDIT-ID PIC X(6).

05 FILLER PIC X(5).

.....

PROCEDURE DIVISION.

MOVE ALL "*" TO EDIT-LINE

WRITE EDIT-LINE

AFTER ADVANCING PAGE

OPEN OUTPUT PAYROLL-EDIT-REPORT

OPEN INPUT TIME-CARD-FILE

.....

افحص بدقة جزء الإجراءات، منطقة السجل المنطقي EDIT- LINE الخاصة بـ REPORT:PAYROLL- EDIT يجب
إلا تعالج أو تستخدم بعد فتح الملف . وكما تبدو الأشياء ستكون للعبارة MOVE ALL "*" TO EDIT- LINE نتائج غير
متوقعة ، ويتسبب WRITE EDIT- LINE في إنهاء البرنامج نهاية غير طبيعية .

مثال ٦ - ٢ :

افرض أن جزء FILE STATUS (القسم الخامس من الفصل الرابع) مستخدم في البرنامج (المصحح) من مثال

(٢-٦).

.....
FILE-CONTROL.

SELECT TIME-CARD-FILE

ASSIGN TO WKLYTIME

FILE STATUS IS TIME-CARD-STATUS

.....

WORKING-STORAGE SECTION.

01 TIME-CARD-STATUS PIC XX.

.....

PROCEDURE DIVISION.

.....

OPEN INPUT TIME-CARD-FILE

IF TIME-CARD-STATUS NOT EQUAL "00"

.....

TIME- CARD- STATUS معرف بأنه منطقة مكونة من ٢ بايت (PIC XX) . بعد تنفيذ عبارة الفتح يجب أن تحتوي هذه المنطقة الثابت غير العددي "00" ، اذا كان التنفيذ صحيحا . أما إذا كان TIME- CARD- STATUS معرفاً بأنه PIC 99 .. فإنه يستخدم على ذلك ثابتاً عددياً (أو ثابتاً استعارياً مكافئاً له) في الاختيار :

IF TIME-CARD-STATUS NOT EQUAL 0 ...

أو

IF TIME-CARD-STATUS NOT EQUAL ZERO ...

٦ - ٣ مدخلات / مخرجات : عبارة الإغلاق

تنتهى عبارة الإغلاق CLOSE التشغيل الملف وذلك (١) بترك موقع الذاكرة المستخدم للذاكرات الاحتياطية . (٢) وإتمام تشغيل عناوين ضرورى الملفات التى لها نمطية (٣) وعدم عمل كل شئ فعلته عبارة الفتح بصفة عامة . لابد من إغلاق كل ملف سبق فتحه قبل إنهاء البرنامج ، ونسيان ذلك يمكن أن يقود إلى أخطاء جسيمة فى بعض النظم . وتكوين عبارة الإغلاق موضح فى شكل (٦ - ٢) .

$$\begin{array}{l} \text{CLOSE file-name-1} \left[\begin{array}{l} \left\{ \begin{array}{l} \text{REEL} \\ \text{UNIT} \end{array} \right\} \left[\begin{array}{l} \text{WITH NO REWIND} \\ \text{FOR REMOVAL} \end{array} \right] \\ \\ \text{WITH} \left\{ \begin{array}{l} \text{NO REWIND} \\ \text{LOCK} \end{array} \right\} \end{array} \right] \\ \\ \text{[file-name-2} \left[\begin{array}{l} \left\{ \begin{array}{l} \text{REEL} \\ \text{UNIT} \end{array} \right\} \left[\begin{array}{l} \text{WITH NO REWIND} \\ \text{FOR REMOVAL} \end{array} \right] \\ \\ \text{WITH} \left\{ \begin{array}{l} \text{NO REWIND} \\ \text{LOCK} \end{array} \right\} \end{array} \right] \dots \end{array}$$

شكل (٦ - ٢)

الصيغة الأساسية للإغلاق

الصيغة الأكثر استخداماً لعبارة الإغلاق هى - ببساطة - الفعل يتبعه اسم ملف واحد ، أو أسماء ملفات . يمكن غلق الملفات بشكل فردى (مع أخذ ذاكرة أقل من الكمبيوتر) أو مع بعضها (مع أستفراق وقت أقل) ، وبأى ترتيب (دون الاعتماد على ترتيب فتح الملفات) ، وإيا كانت الصيغة المستخدمة لعبارة الإغلاق ، إذا حدد FILE STATUS للملف ، الذى يفتح .. فإن عبارة الإغلاق تتسبب فى وضع شفرة حالة فى عنصر بيانات FILE STATUS ، محددة بذلك نجاح أو فشل محاولة الإغلاق .

مثال (٦ - ٤) :

PROCEDURE DIVISION.

.....
OPEN OUTPUT INVENTORY-MASTER-FILE
.....

(use WRITE to add records to INVENTORY-MASTER-FILE)
.....

CLOSE INVENTORY-MASTER-FILE

OPEN INPUT INVENTORY-MASTER-FILE
.....

(use READ to input logical records from INVENTORY-MASTER-FILE)
.....

CLOSE INVENTORY-MASTER-FILE

OPEN I-O INVENTORY-MASTER-FILE
.....

(use READ and REWRITE to update logical records from INVENTORY-MASTER-FILE)
.....

CLOSE INVENTORY-MASTER-FILE

بمجرد إغلاق الملف.. يمكن إعادة فتحه للتشغيل مرة أخرى . وهنا تم فتح ملف للمخرجات ، وأضيفت بعض السجلات ، ثم أغلق الملف . بعد ذلك فتح الملف للمدخلات ، واسترجعت السجلات المنطقية (مثلما يحدث في طباعة تقرير) ، ثم أغلق مرة ثانية . وأخيرا .. فتح الملف مرة أخرى للمدخلات والمخرجات، وتم تجديد بعض السجلات المنطقية ، وأغلق الملف بعد ذلك .

مثال ٦ - ٥ :

PROCEDURE DIVISION.

.....
OPEN I-O EMPLOYEE-MASTER-FILE
.....

CLOSE EMPLOYEE-MASTER-FILE WITH LOCK

عندما يستخدم جزء CLOSED WITH LOCK .. يغلغ الملف، ولا يمكن فتحه أثناء التشغيل الحالى للبرنامج . ويمكن، بالطبع ، فتح الملف بواسطة برنامج آخر ، أو بنفس البرنامج إذا أعيد تنفيذه .

صيغ مصغرة للإغلاق

تستخدم هذه الصيغ مع ملفات الشرائط فقط . وتسمى بكرة الشريط المخزن عليها أكثر من ملف واحد ببكرة متعددة multiple reel (أو شريط متعدد multiple tape أو حجم متعدد multiple volume) . وتحمل عنوان حجم volume label والذي يعطى اسما للبكرة (اسم الحجم) ومعلومات أخرى عن محتويات الشريط . ولكل ملف عنوانان ملف file labels: عنوان أمامي header label في بداية الملف، وعنوان خلفي trailer label في نهايته . ويظهر العنوان الخلفي في حالة تشغيل الملف عكسيا (OPEN INPUT REVERSED) .

وعكسيا ، فالملف متعدد الحجم multivolume file هو ملف مخزن على أكثر من بكرة واحدة (أو أكثر من مجموعة أقراص واحدة) ..

مثال ٦ - ٦ :

CLOSE EMPLOYEE-MASTER-FILE WITH NO REWIND

عندما يغلق ملف شريط ، - عادة - يعاد لف الشريط الى بداية الملف ، وهنا (١) يوجد EMPLOYEE- MASTER- FILE على بكره متعددة الملفات ، ونريد ترك الشريط فى موضع فتح الملف التالى على الشريط ، أو (٢) يعاد فتح الملف EM- PLOYEE- MASTER- FILE للإسخال العكسى INPUT REVERSED ، ويراد تركه موجوداً عند نهايته .

مثال ٦ - ٧ :

PROCEDURE DIVISION.

```

.....
OPEN    OUTPUT    JOB-COSTING-HISTORY-FILE
.....
WRITE JOB-COSTING-HISTORY-RECORD
.....
CLOSE JOB-COSTING-HISTORY-FILE    UNIT
.....
WRITE JOB-COSTING-HISTORY-RECORD
.....
CLOSE JOB-COSTING-HISTORY-FILE

```

ينتج هذا البرنامج ملف JOB- COSTING- HISTORY- FILE وهو متعدد الحجم ، سبق فتحه كملف مخرجات . تستخدم أول عبارة WRITE فى إخراج سجلات إلى الملف . عبارة CLOSE... UNIT (أو المكافئ لها تماما CLOSE... REEL) لاتغلق الملف فعليا ، الا أنها تتأكد تلقائيا أن عبارة WRITE التالية ستخرج سجلا إلى البكرة أو مجموعة الاقراص التالية من الملف (يسمى هذا بمفتاح الحجم volume switch) . لاحظ أنه عندما كتبت كل السجلات المنطقية.. تم أغلاق الملف بطريقة طبيعية .

بالنسبة للملف متعدد الحجم الذى فتح كمداخلات.. تدخل عبارة READ - التى تأتى بعد CLOSE ... UNIT - سجلات من البكرة التالية أو مجموعة الاقراص التالية .

إذا استبدلت عبارة الإغلاق ، فى البرنامج السابق ، بالعبارة التالية :

CLOSE JOB-COSTING-HISTORY-FILE UNIT FOR REMOVAL.

فلن يؤثر نظام التشغيل فقط على مفتاح الحجم فقط، ولكنه يعرف كذلك أنه ليست هناك حاجة إلى الوحدة التى أغلقت مرة أخرى. ويمكن أن يحرر ذلك وحدة المدخلات والمخرجات لاستخدامها فى أى استخدام آخر . إلا أنه يمكن الاتصال بالوحدة المغلقة مرة أخرى، إذا أغلق الملف بطريقة طبيعية ثم أعيد فتحه .

عبارات مدخلات ومخرجات اخرى

بجانب عبارتى الفتح OPEN والإغلاق CLOSE ، توجد عبارة إعادة الكتابة REWRITE ، والتى يمكنك الرجوع اليها فى الفصل الحادى عشر من الكتاب .

٦ - ٤ مدخلات : عبارة القراءة

يمكن استخدام عبارة القراءة مع الملفات التي تم فتحها - بنجاح - كمدخلات او مخرجات فقط . وشكلها العام هو :

READ file-name RECORD [INTO data-name]
[AT END imperative-statement-1 [imperative-statement-2] ...]

يجب أن يكون file- name هو نفسه المستخدم في عبارة SELECT وفي FD للملف . بافتراض الترتيب التتابعى - كما هو الحال دائما - تجعل عبارة READ السجل المنطقى التالى متاحا واقعيا في منطقة السجل المنطقى للملف (المستوى 01 لوصف السجل في FD) . فاذا ما عرفت منطقة FILE STATUS للملف .. تضع عبارة READ محتويات مفتاح الحالة لتحديد نتائج READ .

مثال ٦ - ٨ :

يتطلب تكوين READ استخدام اسم الملف بدلا من اسم السجل المنطقى . وعكس ذلك تماما صحيح بالنسبة لعبارة WRITE (انظر القسم الخامس من هذا الفصل) . وتجنب الخلط .. تذكر شعار الكويل « اقرأ ملفا واكتب سجلا » .
عندما تنفذ READ ولا تكون هناك سجلات منطقية متاحة ، يتحقق شرط نهاية الملف end of file . وتنفذ العبارات الموجودة في جزء AT END وذلك عندما يتحقق شرط نهاية الملف فقط ، والا أهمل الكمبيوتر هذا الجزء . عند الوصول إلى نهاية الملف .. يجب اغلاق الملف والاستجابة قبل محاولة أى تشغيل آخر .

مثال ٦ - ٩ :

.....
ENVIRONMENT DIVISION.

SELECT SALES-FILE ASSIGN TO SALES.

DATA DIVISION.
FILE SECTION.

FD SALES-FILE
BLOCK CONTAINS 0 RECORDS
RECORD CONTAINS 39 CHARACTERS
LABEL RECORDS ARE STANDARD

01 SALES-RECORD.	
05 SALES-RECORD-NAME	PIC X(15).
05 SALES-RECORD-MONTH-1-SALES	PIC S9(4)V99.
05 SALES-RECORD-MONTH-2-SALES	PIC S9(4)V99.
05 SALES-RECORD-MONTH-3-SALES	PIC S9(4)V99.
05 SALES-RECORD-QUOTA	PIC S9(4)V99

WORKING-STORAGE SECTION.

01 WORK-AREAS.	
05 SALES-FILE-END-SW	PIC X.
05 QUARTERLY-TOTAL.	PIC S9(5)V99 COMP-3.

PROCEDURE DIVISION.

OPEN INPUT SALES-FILE
MOVE "F" TO SALES-FILE-END-SW

.....
READ SALES-FILE

AT END

MOVE "T" TO SALES-FILE-END-SW

IF SALES-FILE-END-SW EQUAL "F"

ADD SALES-RECORD-MONTH-1-SALES
SALES-RECORD-MONTH-2-SALES
SALES-RECORD-MONTH-3-SALES

GIVING QUARTERLY-TOTAL

.....

بعد فتح SALES- FILE كمنخالات ، توضع قيمة ابتدائية للمفتاح SALES- FILE- END- SW مساوية "F" (أي خطأ fa lsc) . يحاول بعد ذلك البرنامج قراءة سجل منطقي SALES- FILE .

فإذا لم تتواجد سجلات منطقية ينفذ AT END ، ويوضع "T" (أي صحيح true) كقيمة للمفتاح . فإذا ما تمت قراءة سجل بنجاح فعلا ، يهمل جزء AT END ويظل المفتاح له القيمة "F" . لاحظ أن البرنامج يختبر ما إذا كان SALES- FILE في حالة AT END قبل محاولته تنفيذ ADD . لاحظ كذلك استخدام النقطة في تحديد تعليق لجزء AT END . هذا ضروري ، حيث أنه يفترض أن يستمر الجزء حتى تظهر نقطة .

مثال ٦ - ١٠ :

اعتبر جزء البرنامج التالي :

READ SALES-FILE RECORD

AT END

MOVE "T" TO SALES-FILE-END-SW

MOVE SALES-RECORD-QUOTA TO SALES-REPORT-QUOTA

MOVE "NO" TO SALES-INPUT-ERROR-FLAG

WRITE...

أولاً.. نلاحظ أن المبرمج ، بنسيانه وضع نقطة بعد جزء AT END استخلص MOVE... و MOVE... و WRITE... غير كفؤة قبل نهاية الملف (الترحيل لم يقاوم) . ثانياً.. وربما أقل وضوحاً ، لقد نسي أنه إذا قابلت READ نهاية الملف.. فإن منطقة السجل المنطقي SALES- RECORD ستحتوي على نفايا ، تنقل على ذلك إلى SALES- REPORT- QUOTA : متسببة في مخرجات غير صحيحة وربما في أخطاء أخرى .

لاتقم بتشغيل سجل منطقي بعد تنفيذ AT END... (إلا إذا كان الملف قد أغلق وأعيد فتحه) .

بديل اقرأ ... فى

يتسبب بديل اقرأ ... فى READ... INTO لعبارة ERAD فى نسخ السجل المنطقى الذى يكون مدخلات من منطقة السجل المنطقى فى عنصر بيانات آخر ؛ أى أن :

READ SALES-FILE INTO WORKING-COPY

و

READ SALES-FILE
MOVE SALES-RECORD TO WORKING-COPY

متكافئان تماما . وتستخدم READ... INTO بكثرة لتوفير نسخ من السجلات المنطقية فى مخزن العمل (انظر المسائلتين ٩٧ و ٩٨ من الفصل الخامس) .

ويمكن تطبيق هذا البديل فى كويل 1974 النمطى على سجلات ثابتة الطول فقط (انظر ملحق جـ بالنسبة لكويل 1985 النمطى) .

٦ - ٥ مخرجات : عبارة الكتابة

يمكن استخدام عبارة الكتابة WRITE مع الملفات التى سبق فتحها كملفات مخرجات، أو ملفات مدخلات ومخرجات أو ملفات موسعة . ينقل تنفيذ العبارة محتويات منطقة السجل المنطقى (المستوى 01) إلى المخرجات ، وفى نفس الوقت يغير الاشارة إلى اسم السجل على مستوى 01 إلى منطقة الذاكرة الاحتياطية غير المستخدمة حاليا المصاحبة للملف .

(راجع «الذاكرات الاحتياطية المتعددة» فى القسم الثامن من الفصل الخامس و«التجميع» فى القسم الثالث من الفصل الخامس ، ويمكن استخدام جزء الذاكرة الاحتياطية الجديدة ، والذى يحتوى على نفايا عند هذه النقطة ، فى بناء السجل المنطقى التالى للمخرجات)، ويتبع ذلك ، أن السجل المنطقى الذى كتب تـو لا يمكن الاتصال به بواسطة البرنامج .

الكتابة لملفات غير موجهة للطابع

يكون تكوين عبارة WRITE هنا على النحو التالى :

WRITE logical-record-name [FROM data-name]

يستخدم المبرمج جزء FROM عادة، وعندما يريد عمل كل التشغيل للسجل، مستخدما نسخا للسجل المنطقى من مخزن العمل . (تناظر READ... FROM من القسم الرابع من هذا الفصل) .

WRITE INVOICE-TO-BE-PAID-RECORD
FROM WS-INVOICE-TO-BE-PAID-REC

لها نفس ترجمة الآلة تماما مثل :

MOVE WS-INVOICE-TO-BE-PAID-REC
TO INVOICE-TO-BE-PAID-RECORD
WRITE INVOICE-TO-BE-PAID-RECORD

يجب أن يكون السجل المنطقي logical-record-name اسم العنصر الموجود على مستوى 01 في FD للملف .
 عند استخدام جزء FROM ، عادة ما يعرف "data-name" في مخزن العمل، وإذا تم تعريف منطقة FILE STATUS للملف.. فإن WRITE تضع شفرة مكونة من 2 بايت في منطقة مفتاح الحالة لتحديد نتيجة تنفيذ WRITE

مثال ٦ - ١١ :

FD PENDING-INVOICE-FILE
 BLOCK CONTAINS 0 RECORDS
 RECORD CONTAINS 32 CHARACTERS
 LABEL RECORDS ARE STANDARD

01 PENDING-INVOICE-RECORD PIC X(32);

FD PAYABLE-INVOICE-FILE
 BLOCK CONTAINS 0 RECORDS
 RECORD CONTAINS 16 CHARACTERS
 LABEL RECORDS ARE STANDARD

01 PAYABLE-INVOICE-RECORD PIC X(16);

WORKING-STORAGE SECTION.

01 WS-PENDING-INVOICE REC.
 05 WS-PENDING-CODE PIC X.
 05 WS-PENDING-VENDOR-ID PIC X(7).
 05 WS-PENDING-INVOICE-NUMBER PIC X(4).
 05 WS-PENDING-INVOICE-DATE PIC 9(6).
 05 WS-PENDING-INVOICE-DUE-DATE PIC 9(6).
 05 WS-PENDING-INVOICE-AMOUNT PIC S9(5)V99 COMP-3.
 05 WS-PENDING-INVOICE-DISCOUNT PIC S9(5)V99 COMP-3.

01 WS-PAYABLE-INVOICE-REC.
 05 WS-PAYABLE-CODE PIC X.
 05 WS-PAYABLE-VENDOR-ID PIC X(7).
 05 WS-PAYABLE-INVOICE-NUMBER PIC X(4).
 05 WS-PAYABLE-AMOUNT PIC S9(5)V99 COMP-3.

PROCEDURE DIVISION.

OPEN INPUT PENDING-INVOICE-FILE
 OUTPUT PAYABLE-INVOICE-FILE

 READ PENDING-INVOICE-FILE
 INTO WS-PENDING-INVOICE-REC
 AT END
 MOVE "T" TO WS-END-OF-FILE-SW

IF WS-END-OF-FILE-SW NOT EQUAL "T"
 MOVE WS-PENDING-CODE TO WS-PAYABLE-CODE
 MOVE WS-PENDING-VENDOR-ID TO WS-PAYABLE-VENDOR-ID
 MOVE WS-PENDING-INVOICE-NUMBER TO WS-PAYABLE-INVOICE-NUMBER
 SUBTRACT WS-PENDING-INVOICE-DISCOUNT

FROM WS-PENDING-INVOICE-AMOUNT
GIVING WS-PAYABLE-AMOUNT

WRITE PAYABLE-INVOICE-RECORD FROM WS-PAYABLE-INVOICE-REC

.....
CLOSE PENDING-INVOICE-FILE
PAYABLE-INVOICE-FILE
.....

يوضح هذا البرنامج استخدام نسخ مخزن العمل للسجلات المنطقية .

يوصف PAYABLE- INVOICE- RECORD و PENDING- INVOICE- RECORD أكثر بصورة (33) PIC X و (16) PX على التوالي، حيث أنه لن يجرى عليهما تشغيل في منطقة السجل المنطقي للذاكرة الاحتياطية . ويستخدم READ و (16) PX على التوالي، حيث أنه لن يجرى عليهما تشغيل في منطقة السجل المنطقي للذاكرة الاحتياطية . ويستخدم MOVE و SUBTRACT في إعداد السجل المنطقي في WS- PAYABLE- IN- VOICE- REC . ثم تستخدم عبارات MOVE و SUBTRACT في إعداد السجل المنطقي في WS- PAYABLE- IN- VOICE- REC . ثم تستخدم بعد ذلك العبارة WRITE PAYABLE- INVOICE- RECORD FROM WS- PAYABLE- INVOICE- RECORD في نقل هذا السجل من مخزن العمل إلى منطقة السجل المنطقي، في الذاكرة الاحتياطية ونقله إلى المخرجات .

بعد عبارة WRITE... لا يحتوي PAYABLE- INVOICE- RECORD على البيانات التي أخرجت على التو، بل أنه يشير إلى جزء الذاكرة الاحتياطية التالي الفارغ والمتاح ، والذي يمكن استخدامه في إعداد السجل المنطقي الجديد . إلا أن WS- PAYABLE- INVOICE- REC لا يزال محتويا على السجل المنطقي الذي كتب على التو ، ويشتمل هذا على ميزة أخرى؛ تتعلق بإجراء تشغيل كل السجلات في مخزن العمل .

كما في استخدام READ... INTO يجب إلا تستخدم WRITE... FROM مع سجلات متغيرة الأطوال على الإطلاق (انظر ملحق د لكويل 1985 النمطي) . يتطلب منهج مخزن العمل لمثل هذه السجلات زوجا من عبارات MOVE و WRITE :

MOVE WS-PAYABLE-INVOICE-REC TO PAYABLE-INVOICE-RECORD
WRITE PAYABLE-INVOICE-RECORD

الكتابة لملفات موجهة للطابع

يبين تكوين عبارة WRITE في هذه الحالة في شكل (٦ - ٣) .

WRITE record-name [FROM identifier]
[{ BEFORE } ADVANCING { { identifier-2 } [LINE]
{ integer } [LINES]
{ mnemonic-name }
[PAGE] }]
[AT { END-OF-PAGE } imperative-statement-1 [imperative-statement-2] ...]
EOP

شكل (٦ - ٣)

مثال (٦ - ١٢) :

FD TIME-CARD-FILE
BLOCK CONTAINS 0 RECORDS
RECORD CONTAINS 33 CHARACTERS
LABEL RECORDS ARE STANDARD

01 TIME-CARD-RECORD PIC X(33).

FD TIME-CARD-EDIT-REPORT
RECORD CONTAINS 132 CHARACTERS
LABEL RECORDS ARE OMITTED

01 TIME-CARD-EDIT-LINE PIC X(132).

WORKING-STORAGE SECTION.

01 WS-TIME-CARD-REC.
05 WS-TIME-CARD-ID PIC X(6).
05 WS-TIME-CARD-NAME PIC X(20).
05 WS-TIME-CARD-DEPARTMENT PIC X(4).
05 WS-TIME-CARD-HOURS PIC S99V9.

01 WS-TIME-EDIT-LINE.
05 WS-TIME-EDIT-ID PIC X(6).
05 FILLER PIC X(4) VALUE SPACES.
05 WS-TIME-EDIT-NAME PIC X(20).
05 FILLER PIC X(10) VALUE SPACES.
05 WS-TIME-EDIT-DEPARTMENT PIC X(4).
05 FILLER PIC X(6) VALUE SPACES.
05 WS-TIME-EDIT-HOURS PIC ZZ.9--.
05 FILLER PIC X(77) VALUE SPACES.

01 NUMBER-TO-SKIP PIC S9(3) COMP-3.

PROCEDURE DIVISION.

OPEN INPUT TIME-CARD-FILE
OUTPUT TIME-CARD-EDIT-REPORT

READ TIME-CARD-FILE
INTO WS-TIME-CARD-REC
AT END...

MOVE WS-TIME-CARD-ID TO WS-TIME-EDIT-ID
MOVE WS-TIME-CARD-NAME TO WS-TIME-EDIT-NAME
MOVE WS-TIME-CARD-DEPARTMENT TO WS-TIME-EDIT-DEPARTMENT
MOVE WS-TIME-CARD-HOURS TO WS-TIME-EDIT-HOURS

WRITE TIME-CARD-EDIT-LINE FROM WS-TIME-EDIT-LINE
BEFORE ADVANCING 3 LINES

يتسبب WRITE... BEFORE ADVANCING 3 LINES في طباعة المحتويات الحالية لـ TIME-EDIT-LINE عن طريق TIME-CARD-EDIT-LINE ، وذلك عندما يكون الورق موجودا في الطابع . بعد طباعة السطر .. يتقدم الورق ثلاثة سطور للامام . وعلى هذا إذا نفذت WRITE مرات ومرات .. فتكون النتيجة ترك سطرين فارغين بين السطور المطبوعة .

يستغل المثال منهج مخزن العمل ، من حيث استخدام جزء VALUE فى وضع مناطق FILLER (التى تفصل الحقول على الورق المطبوع) على أنها فراغات . فإذا بنينا TIME- EDIT- LINE فى قسم الملفات (حيث لايسمح باستخدام VALUE) .. فيجب أن نستخدم عبارة MOVE فى نقل فراغات SPACES الى منطقة السجل المنطقى قبل الانتقال إلى المعلومات المراد طباعتها .

إذا لم يوجد الجزء BEFORE ADVANCING 3 LINES فى البرنامج السابق ، يعمل الجزء التقليدى AFTER ADVANCING 1 LINE . وعلى هذا .. لاتوجد سطور فارغة بين السطور المطبوعة .

مثال ٦ - ١٣ :

لتوضيح استخدام هذا الجزء

WRITE ... BEFORE/AFTER ADVANCING identifier-2 LINES

افرض - كمثال - أننا عرفنا اسم بيانات NOMBRE- TO - SKIP ، والذي يجب أن يكون له PICTURE عددى بدون علامة عشرية . عند ذلك :

```
.....
MOVE 3 TO NUMBER-TO-SKIP
.....
WRITE TIME-CARD-EDIT-LINE
FROM WS-TIME-EDIT-LINE
AFTER ADVANCING NUMBER-TO-SKIP LINES
```

يمكن أن تكون محتويات NOMBRE- TO - SKIP أى مدخلات صحيحة اختيارية يدخلها البرنامج، أو تنتج من حسابات مبرمجة . يتسبب WRITE... AFTER ADVANCING فى تقويم الورق بعدد السطور المحدد ، ثم إخراج السطر المراد طباعته .

مثال ٦ - ١٤ :

ينقل بديل BEFORE/AFTER ADVANCING PAGE الورق إلى بداية الصفحة التالية، وإذا حدد جزء الخطية LINAGE فى FD للملف ، الورق يوضع عند بداية جسم الصفحة للصفحة المنطقية التالية (انظر القسم رقم 7 من الفصل الخامس). وإذا .. تم وضع الورق عند القناة 1 لآلية التحكم فى عربة الطابع (والتي تتناظر بصفة عامة السطر 1 فى الصفحة التالية، انظر القسم الرابع من الفصل الرابع) .

مثال ٦ - ١٥ :

يسمح جزء BEFORE/AFTER ADVANCING mnemonic- name بوضع الورق فى قناة تحكم العربة المعرفة، فى مقطع الأسماء الخاصة من جزء الأوساط (راجع قسم ٤ - ٤) .

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
C01 IS TOP-OF-FORM
C02 IS HEADING-AREA
C05 IS TOTAL-AREA

.....
PROCEDURE DIVISION.
WRITE REPORT-LINE
FROM WS-PAGE-NUMBER-DATE-LINE
AFTER ADVANCING TOP-OF-FORM
WRITE REPORT-LINE
FROM WS-HEADING-LINE
AFTER ADVANCING HEADING-AREA
WRITE REPORT-LINE
FROM WS-TOTAL-LINE
AFTER ADVANCING TOTAL-AREA

هنا .. يقدم AFTER ADVANCING TOP- OF- FORM الورق إلى قناة تحكم العربية رقم 1 ، ثم تطبع المحتويات الحالية لـ WS- PAGE- NUMBER- DATE- LINE .. إلخ. وقبل السماح بتنفيذ البرنامج ، يجب أن يعد مشغل الكمبيوتر الطابع بحيث تكون قنوات التحكم في العربية المستخدمة - بواسطة البرنامج - متفقة مع الأجزاء المناسبة من الصيغ المستمرة : فإذا لم يحدث ذلك .. فلن تكون مسافات الطباعة مضبوطة .

إن استخدام قنوات التحكم في العربية المعرفة في مقطع الاسماء الخاصة واستخدام جزء الخطية LINAGE في FD مستقلان عن بعضهما البعض .

في شكل (٦ - ٣) "AT END- OF- PAGE" و "AT EOP" متكافئان . ويمكن استخدامهما عندما يكون جزء LINAGE محددًا في FD فقط .. في هذه الحالة ، يتسبب مترجم الكويل في حفظ برنامج الهدف تلقائيًا لصيانة LINAGE- COUNTER الذي يعد كل السطور المطبوعة أو المتروكة في صفحة منطقية . وكلما بدأت صفحة منطقية جديدة .. يعاد وضع القيمة 1 في LINAGE- COUNTER بصورة تلقائية .

تنفذ العبارات المحددة في جزء AT END- OF- PAGE عندما يتسبب تنفيذ WRITE BEFORE/AFTER ADVANCING في أن يتعدى LINAGE- COUNTER (١) القيمة المحددة في ... WITH FOOTING AT (يختلف حجم جسم الصفحة عما هو محدد بواسطة LINAGE IS) أو (٢) حجم جسم الصفحة .

• **الحالة الأولى :** تنفذ عبارة WRITE كما هو الحال المعتاد، وتنفذ العبارات في جزء AT END- OF- PAGE، ثم يستأنف التنفيذ المعتاد العبارات .

• **الحالة الثانية :** إذا تحدد BEFORE ADVANCING .. يطبع السطر ، ثم يتقدم الورق إلى بداية جسم الصفحة المنطقية التالية ، وإذا تحدد AFTER ADVANCING .. يتقدم الورق إلى بداية جسم الصفحة المنطقية التالية ، ثم يطبع السطر، وتنفذ بعد ذلك عبارات AT END- OF- PAGE .

مثال ٦ - ١٦ :

تسمى الحالة الثانية بالسريان الزائد التلقائي للصفحة automatic page overflow . افترض انه بالنسبة إلى RE-
: PORT- FILE

LINAGE IS 60
WITH FOOTING AT 53
LINES AT TOP 3
LINES AT BOTTOM 3

وأن الورق موجود عند السطر الثاني والخمسين داخل جسم الصفحة حالياً . ويصمم تنفيذ مايلي :

WRITE REPORT-LINE
FROM WS-CHECK-REQUEST-EDIT-LINE
AFTER ADVANCING 10 LINES
AT END-OF-PAGE
MOVE WS-RUNNING-TOTAL TO WS-OUTPUT-RUNNING-TOTAL
WRITE REPORT-LINE FROM WS-RUNNING-TOTAL-LINE
AFTER ADVANCING 5 LINES

لطباعة إجمالي جار running total في منطقة النهاية ، في نهاية جسم الصفحة . إلا أنه حيث إن AFTER
ADVANCING 10 LINES يتسبب في سريان زائد للصفحة (١٠ + ٥٢ = ٦٢) ، والتي تتعدى حجم جسم الصفحة التي لها
LINAGE 60 فيحدث مايلي :

(١) يتقدم الورق إلى أول سطر طباعة في الصفحة المنطقية التالية .

(٢) تطبع محتويات WS- CHECK- REQUEST- EDIT- LINE .

(٣) نظر لحديث سريان زائد تلقائي للصفحة .. فإن جزء AT END- OF- PAGE يتسبب في تقديم الورقة خمسة أسطر

أخرى ، وتطبع بعدها محتويات WS- RUNNING- TOTAL- LINE .

وعلى هذا .. يظهر سطر الإجمالي الجارى بالقرب من بداية جسم الصفحة ، مختلطاً مع أسطر التقرير المعتادة - كوضع
غير مرغوب فيه .

مثال ٦ - ١٧ :

يمكن أن يحذف البرمج صعوبات أجزاء AT END- OF- PAGE ، وذلك بالتنسيق المحتمل لقيمة BEFORE/AFTER
ADVANCING ، مع قيمة WITH FOOTING AT ، وجزء AT END- OF- PAGE .

اجعل جزء LINAGE كما هو في مثال (٦ - ١٦) . افترض كذلك أن في بداية كل جسم صفحة يطبع عنواناً يترك الورق
موضوعاً عند السطر رقم 10 من جسم الصفحة . أخيراً ، افترض أنه مطلوب طباعة سطور تقرير ، مع وجود سطرين فارغين
بين السطور المطبوعة ، أعتبر WRITE التالية :

WRITE REPORT-LINE
FROM WS-CHECK-REQUEST-EDIT-LINE
AFTER ADVANCING 3 LINES
AT END-OF-PAGE
MOVE WS-RUNNING-TOTAL TO WS-OUTPUT-RUNNING-TOTAL
WRITE REPORT-LINE
FROM WS-RUNNING-TOTAL-LINE
AFTER ADVANCING 5 LINES

WRITE REPORT-LINE
FROM WS-FIRST-TITLE-LINE
AFTER ADVANCING PAGE
WRITE REPORT-LINE
FROM WS-SECOND-TITLE-LINE
AFTER ADVANCING 9 LINES

عندما تنفذ عبارة WRITE هذه لأول مرة فإنها تجعل الورق موضوعا عند نهاية العنوان (السطر العاشر) . وعلى هذا .. يصبح أول سطر في التقرير الثالث عشر (AFTER ADVANCING 3 LINES) ، ويصبح السطر الثاني في السطر السادس عشر... ويقع السطر الرابع عشر من التقرير في السطر ٥٢ . يؤدي التنفيذ الخامس عشر لعبارة WRITE إلى حدوث مايلي :

(١) طباعة السطر الخامس عشر من التقرير في السطر رقم ٥٥ من الصفحة .

(٢) حيث إن LINAGE- COUNTER (٥٥ الآن) تعدى حد منطقة النهاية (WITH FOOTING AT 53) ... يتسبب جزء AT END- OF- PAGE في تقديم الورقة الى السطر رقم ٦٠ . (AFTER ADVANCING 5 LINES) ، وتطبع محتويات WS- RUNNING- TOTAL- LINE على السطر رقم ٦٠ ، (آخر سطر في جسم الصفحة) ، كما يكتب أول سطر من عنوان الصفحة التالية على السطر رقم 1 من جسم الصفحة التالية (AFTER ADVANCING PAGE) ، وأخيرا ، تكتب بقية العنوان على السطر العاشر (AFTER ADVANCING 9 LINES) من السطر رقم 1 .

لاحظ عدم قيام جزء AT END- OF- PAGE بطبع الإجمالي الجارى في منطقة النهاية من كل صفحة فقط ، لكنه يقدم الورق إلى الصفحة المنطقية التالية ويطبع العنوان . يترك الورق موضوعا عند السطر العاشر ، بحيث تبدأ الدورة مرة أخرى .

مثل AT END (القسم الرابع من هذا الفصل) فيفترض أن تستمر AT END- OF- PAGE حتى تظهر نقطة .

مثال ٦ - ١٨ :

READ CHECK-REQUEST-FILE
AT END
MOVE "T" TO CHECK-REQUEST-END-SW
MOVE CHECK-REQUEST-VENDOR-ID TO WS-VENDOR-ID
MOVE CHECK-REQUEST-AMOUNT TO WS-AMOUNT
WRITE CHECK-REQUEST-EDIT-LINE
FROM WS-EDIT-LINE
AFTER ADVANCING 2 LINES
AT END-OF-PAGE
WRITE CHECK-REQUEST-EDIT-LINE
FROM CHECK-REQUEST-EDIT-TITLE
AFTER ADVANCING PAGE
D 1 TO NUMBER-CHECK-REQUESTS

يتطلب جزء البرنامج هذا نقطة مرتبة بعد " T " MOVE ، ونقطة مرتبة أخرى بعد AFTER ADVANCING PAGE . كما تعنى.. فإن الترحيل لايعنى شيئاً، يستمر جزء AT END حتى نهاية الجزء . وعلى هذا .. فإنه يشمل أول عبارة WRITE . هي عبارة شرطية ؛ لأنها تحتوى على جزء AT END- OF- PAGE ، الذى يتفد تحت شروط معينة . وعلى هذا يشمل AT END خطأ تكوينياً ؛ لأنه يجب ألا يحتوى إلا على عبارات أمرية غير شرطية (انظر القسم الرابع من هذا الفصل السادس) .

٦ - ٦ مدخلات : عبارة القبول

تستخدم عبارة القبول ACCEPT لمدخلات مخصصة معينة .

قبول معلومات من نظام التشغيل

التكوين لنقل معلومات نظام تشغيل معينة إلى برنامج هدف الكويل، هو :

ACCEPT identifier FROM $\left\{ \begin{array}{l} \text{DATE} \\ \text{DAY} \\ \text{TIME} \end{array} \right\}$

حيث identifier هو اسم بيانات، معرف بطريقة مناسبة فى جزء البيانات .

مثال ٦ - ١٩ :

- WORKING-STORAGE SECTION.
01 SAMPLE-ACCEPTED-DATE PIC 9(6).

PROCEDURE DIVISION.
ACCEPT SAMPLE-ACCEPTED-DATE FROM DATE

يضع نظام التشغيل التاريخ الحالى فى عنصر البيانات SAMPLE- ACCEPTED- DATE. والصورة المناسبة للتاريخ DATE هي (6) PIC9. وتكون صورة التاريخ yymmdd، حيث yy هي السنة، mm هو الشهر، dd هو اليوم، أى ان ١٢ فبراير تمثل على النحو التالى 830212.

- 01 SAMPLE-ACCEPTED-DAY PIC 9(5).
ACCEPT SAMPLE-ACCEPTED-DAY FROM DAY

يضع نظام التشغيل التاريخ الميلادى اليومى julian date فى عنصر البيانات SAMPLE- ACCEPTED- DAY والصورة المناسبة لليوم DAY ، هي (5) PIC 9 . ويمثل التاريخ الميلادى على هيئة عدد من خمسة أرقام فى الصورة :yyddd حيث yy هي السنة ddd هو اليوم (حيث ترقيم الأيام من ١ إلى ٣٦٥، أو من ١ إلى ٣٦٦ فى السنة الكبيسة) . ويمثل ١٢ فبراير ١٩٨٣ على النحو التالى 83043 :

- 01 SAMPLE-ACCEPTED-TIME PIC 9(8).
ACCEPT SAMPLE-ACCEPTED-TIME FROM TIME

يضع نظام التشغيل الوقت من اليوم في عنصر البيانات SAMPLE- ACCEPTED- TIME ، والصورة المناسبة للوقت TIME هي PIC 9 (8). ويمثل الوقت من اليوم على هيئة عدد من ثمانية أرقام في الصورة HHmmsshh ، حيث HH : هي الساعة (من ٠ إلى ٢٣) ، وMM في الدقائق (من ٠ إلى ٥٩) ، SS هي الثواني (من ٠ إلى ٥٩) ، وhh هي جزء من المائة من الثانية (من ٠ إلى ٩٩) . وعلى هذا .. تظهر الساعة 24 : 3 بعد الظهر في الصورة التالية 15240000 .

قبول بيانات من SYSIN ، أو من نهاية طرفية يعمل عليها مشغل الكمبيوتر

يمكن استخدام الصيغة الأخرى لعبارة ACCEPT في إدخال كم بسيط من المعلومات من ملف نظام خاص أو ، الأكثر اعتيادا ، من نهاية طرفية يعمل عليها مشغل الكمبيوتر . والتكوين هو كما يلي :

ACCEPT identifier [FROM special-name]

يجب أن يعرف special- name في مقطع الأسماء الخاصة من جزء الأوساط (القسم الرابع من الفصل الرابع). وتختلف التفاصيل الدقيقة لصيغة ACCEPT هذه من نظام لآخر ، إلا أن المحتويين المقبولين في قسم الأسماء الخاصة في كويل IBM OS/VS لعبارة ACCEPT ، هما : CONSOLE و SYSIN :

SPECIAL-NAMES.

CONSOLE IS OPERATOR-MESSAGE-DEVICE
SYSIN IS BATCH-CONTROL-CARD

كلمة CONSOLE .. هي إحدى كلمات الكويل المحجوزة من كويل IBM ، والتي تحدد النهاية الطرفية التي يعمل عليها مشغل الكمبيوتر ، التي تستخدم في الاتصال بنظام التشغيل وبرنامج التنفيذ الأخرى من خلال نظام التشغيل . كلمة SYSIN هي كلمة محجوزة في نظم IBM ، تحدد ملف خاص له سجلات أطوالها ٨٠ بايت؛ فإذا أراد المبرمج استخدام ملف SYSIN .. فيجب تقديم عبارة لغة تحكم عمل لتعريفه . ويصفة عامة.. يجب تجنب استخدام ACCEPT في الإدخال من ملف SYSIN إلا إذا كانت المعلومات المشمولة صغيرة جدا . من الأفضل إعداد ملف معتاد بعبارة SELECT ، واستخدام FD و OPEN و READ .

مثال (٦ - ٢٠) :

SPECIAL-NAMES.

CONSOLE IS OPERATOR-MESSAGE-DEVICE

WORKING-STORAGE SECTION.

01 NUMBER-OF-LABELS-ACROSS PIC 9 DISPLAY.

PROCEDURE DIVISION.

ACCEPT NUMBER-OF-LABELS-ACROSS FROM OPERATOR-MESSAGE-DEVICE

عندما تطبع العناوين البريدية أو الشيكات أو أى صيغ صغيرة أخرى باستخدام الكمبيوتر ، فيمكن زيادة سرعة الطباعة بطباعة العديد من الصيغ فى السطر الواحد . ويوضح الجزء السابق من برنامج كيف يتمكن مشغل الكمبيوتر من تحديد عدد العناوين التى يجب أن تطبع فى الصفحة فى هذا التنفيذ الخاص للبرنامج . عندما تنفذ عبارة ACCEPT ، يتوقف تنفيذ بقية البرنامج مؤقتا حتى يكتب مشغل الكمبيوتر المعلومات المطلوبة مستخدما لوحة مفاتيح النهاية الطرفية . عند ذلك توضع المعلومات فى عنصر البيانات المحدد بواسطة عبارة ACCEPT ، وهو NUMBER- OF- LABELS- ACROSS . لأن البيانات المقبولة تأتى من نهاية طرفية ملحق بها شاشة، أو ملحق بها طابع ، فيفضل استخدام DISPLAY . يجب أن يكتب مشغل الكمبيوتر المعلومات طبقا للصورة PICTURE الموجودة فى برنامج الكويل ، وإلا فسوف تحدث أخطاء .

مثال ٦ - ٢١ :

من ضمن الاستخدامات القليلة العملية لعبارة ACCEPT من SYSIN ، هى تقديم مدخلات بطاقة تحكم . بطاقات التحكم هى control cards هى سجلات (عادة ماتكون مثقبة فى بطاقات) تحدد بدائل البرنامج التى تعمل بالنسبة لتشغيل الكمبيوتر الحالى . افترض أن برنامج العناوين البريدية يرتب العناوين طبقا للرمز البريدى أو طبقا للاسم ، وأنه يمكن أن يشتمل أو لا يشتمل على العاملين بالشركة .

SPECIAL-NAMES.

SYSIN IS OPTION-INPUT-DEVICE

WORKING-STORAGE SECTION.

01 OPTIONS-FOR-THIS-RUN.

05 TYPE-OF-SORT PIC X(4)
05 INCLUDE-EMPLOYEES PIC X(3).

PROCEDURE DIVISION.

ACCEPT OPTIONS-FOR-THIS-RUN FROM OPTION-INPUT-DEVICE

يمكن أن تحتوى بطاقة التحكم على مايلى :

<u>NAMENOB</u>	or	<u>ZIPbYES</u>
X(4) X(3)		X(4) X(3)

للتسبب في أن البرنامج يرتب العناوين طبقاً للاسم مع حذف العاملين من البريد ، أو للترتيب طبقاً للرقم البريدي مع شمول العاملين وذلك على التوالي . يدخل كل تنفيذ لعبارة ACCEPT سجل منطقي جديد من ملف SYSIN . لاحظ أنه بالرغم من أن طول سجلات SYSIN هو ٨٠ بايت ، إلا أنه ليس هناك حاجة لحساب بايت إضافي .
 73 = (4+3) - 80 في وصف OPTIONS- FOR- THIS- RUN .

٦ - ٧ مخرجات : عبارة العرض

تصمم عبارة العرض DISPLAY لإنتاج مخرجات مطبوعة محدودة الحجم على ملفات نظام خاصة أو على النهاية الطرفية التي يستخدمها مشغل الكمبيوتر . ومثل نظيرتها ACCEPT .. فإن التفاصيل الدقيقة لعبارة العرض تتغير من نظام لآخر وتكونها في كابل IBM OS/VS هو كما يلي :

DISPLAY { identifier-1 } [identifier-2] ... [UPON special-name]
 [literal-1] [literal-2]

حيث يجب تعريف special- name في مقطع الأسماء الخاصة (القسم الرابع من الفصل الرابع) . والمحتويات المقبولة في مقطع الأسماء الخاصة لعبارة DISPLAY هي :

SPECIAL-NAMES.
 CONSOLE IS ...
 SYSOUT IS ...

وهنا CONSOLE هي نفسها كما ذكرت في القسم السادس من هذا الفصل، و SYSOUT هي المخرجات المناظرة لـ SYSIN.

مثال ٦ - ٢٢ :

يمكن استخدام DISPLAY (بالاتصال مع ACCEPT) في التداخل مع مشغل الكمبيوتر، وذلك عن طريق النهاية الطرفية لمشغل الكمبيوتر .

SPECIAL-NAMES.
 CONSOLE IS OPERATOR-MESSAGE-DEVICE
 WORKING-STORAGE SECTION.

01 WS-OPERATOR-REPLIES.
 05 WS-NUMBER-OF-LABELS-ACROSS PIC 9.
 05 WS-EMPLOYEES-INCLUDED PIC X(3).

PROCEDURE DIVISION.

DISPLAY "PLEASE SET UP PRINTER FOR MAILING LABELS"
 UPON OPERATOR-MESSAGE-DEVICE

DISPLAY "WHEN READY, ENTER NUMBER OF LABELS ACROSS (1 DIGIT)"
 UPON OPERATOR-MESSAGE-DEVICE

ACCEPT WS-NUMBER-OF-LABELS-ACROSS
 FROM OPERATOR-MESSAGE-DEVICE

DISPLAY "EMPLOYEES INCLUDED? ('YES' OR 'NO')"
 UPON OPERATOR-MESSAGE-DEVICE

ACCEPT WS-EMPLOYEES-INCLUDED
 FROM OPERATOR-MESSAGE-DEVICE

الملقنات prompts المطبوعة على النهاية الطرفية المتاحة لمشغل الكمبيوتر توجه المشغل لتنفيذ أنشطة معينة أو ترشده لإدخال معلومات لعبارة ACCEPT .

ويجب أن تتذكر أنه عندما يعرض DISPLAY ثابتاً غير عددي (مثل الملقنات سالفة الذكر) على وحدة مخرجات تشتمل على سطور.. طول السطر 100 رمز مثلاً : فتنجز المخرجات إلى وحدات من 100 رمز ، مع حدوث التجزئة في أى موقع بعد المائة رمز . وفي حالة الثوابت الاستعارية.. يطبع رمز واحد (المخرجات من DISPLAY SPACES هي فراغ واحد فقط) . وتحول DASPLAY كلاً من COMP، أو COMP-3 بصورة تلقائية إلى DISPLAY قبل الطباعة . وفي كويل IBM OS/VS ، تحذف إشارات الموجب تلقائياً، أما إشارات السالب فلا تحذف (انظر جدول ٥ - ١) لاتطبع العلامة العشرية المعرفة بصورة V.

مثال ٦ - ٢٢ :

استخدام آخر أساس لعبارة 'DISPLAY هو في أسطر التصحيح (انظر القسم الثاني من الفصل الرابع).

```

SPECIAL-NAMES.
D      SYSOUT IS DEBUGGING-DEVICE

PROCEDURE DIVISION.
D      DISPLAY "STARTING VALUE OF WS-NUMBER-OF-CLIENTS IS "
D      WS-NUMBER-OF-CLIENTS
D      UPON DEBUGGING-DEVICE
.....
D      DISPLAY "WHEN ID INVALID, EMPLOYEE NAME IS "
D      EMPLOYEE-MASTER-NAME
D      UPON DEBUGGING-DEVICE

```

عندما يحدد مقطع كمبيوتر المصدر : WITH DEBUGGING' MODE، تترجم اسطر التصحيح (التي لها الحرف D في العمود السابع) إلى برنامج هدف متسببة في معلومات مفيدة عن محتويات عناصر البيانات التي تطبع في ملف SYSOUT. لاحظ أن الثوابت غير العددية تشمل فراغا اضافيا (قبل علامة التنصيص التي تنهيها) لفصل العناصر في سطر DISPLAY، ولاتقدم DASPLAY فراغات فاصلة بصورة تلقائية.

مثال ٦ - ٢٤ :

الاستخدام الثالث الرئيسي لعبارة : DISPLAY هو في طباعة رسائل خطأ أو رسائل استثناءات علي ملف طباعة SYS-OUT، بحيث أنه يمكن شد انتباه الشخص إلى شروط الخطأ والاستثناءات المكتشفة أثناء تنفيذ البرنامج. (يجب الا يستخدم هذا الاستخدام عندما يكون متوقعا حدوث عدد معنوي للأخطاء وللاستثناءات. وعلى هذا، فعادة مايتم تشغيل المدخلات المكتوبة بلوحة المفاتيح بواسطة برنامج تنقيح edit program، والذي تكون وظيفته الرئيسية اكتشاف الأخطاء في بيانات المدخلات وطباعة تقرير خطأ معطيا معلومات كافية بحيث يمكن تصحيح الأخطاء. في هذه الحالات، يجب أن تطبع معلومات الأخطاء في ملف معتاد، والذي يعد بواسطة SELECT , FD .. إلخ).

```

SPECIAL-NAMES.
  SYSOUT IS ERROR-MESSAGE-LOG
  .....
PROCEDURE DIVISION.
  .....

OPEN INPUT CHECK-REQUEST-FILE
IF CHECK-REQUEST-STATUS-CODE NOT EQUAL "00"
  DISPLAY "UNABLE TO OPEN CHECK REQUEST FILE"
  UPON ERROR-MESSAGE-LOG

```

إذا لم يستخدم جزء UPON في عبارة DISPLAY فالإجراء التقليدي في كويل IBM OS/VS هو:

DISPLAY ... UPON SYSOUT

٦ - ٨ تشغيل : عبارة النقل

فعل النقل MOVE هو فعل مضلل، فتأثير النقل هو نسخ copy محتويات احد عناصر البيانات في عنصر بيانات آخر أو في عناصر بيانات أخرى. لا يتغير الحقل الراسل sending field عند تنفيذ عبارة النقل. وتستبدل المحتويات الاصلية في الحقل المستقبل recieving field بمحتويات الحقل الراسل. تتسبب عبارة النقل كذلك في تحويل البيانات من حالة استخدام USAGE الحقل المستقبل، إذا ما اختلفت الحالتان، أكثر من هذا، إذا اختلف طول الحقل الراسل عن طول الحقل المستقبل، فتتسبب عبارة النقل في الحذف أو في ملا فراغات أو أصفار في الحقل المستقبل. أخيراً، تتسبب عبارة النقل في تنقيح edit البيانات العديدة إذا كانت صورة PICTURE الحقل المستقبل محتوية على رموز تنقيح.

وعبارة النقل لها التكوين التالي:

MOVE { identifier-1
literal } **TO** identifier-2 [identifier-3] ...

(وهناك تكوين بديل لن يعالج في هذا الكتاب ، الا انه مذكور في مثال ٦ - ٢١) .

مثال ٦ - ٢٥ :

بمعرفة مايلي :

```

01 SAMPLE-MOVE-ITEMS.
05 A   PIC X(5).
05 B   PIC X(5).
05 C   PIC X(3).
05 D   PIC X(7).

```

افرض أن الحقل A يحتوى على "HELLO" ، وعلى هذا.. فإن :

- **MOVE A TO B**

تنسخ محتويات عنصر البيانات A (identifier-1) فى عنصر البيانات B (identifier-2) . وحيث أن A و B لهما نفس الصورة والاستخدام.. فإن كلاً منهما يحتوى على HELLO بعد تنفيذ عبارة النقل . وتفقد المحتويات الأصلية للحقل B .

- **MOVE A TO C**

تنسخ محتويات العنصر A فى العنصر C، وحيث أن صورة العنصر C هي PIC X(3) يحذف جزء من HELLO من ناحية اليمين : بعد تنفيذ عبارة النقل.. يحتوى A على HELLO ، بينما يحتوى C على HEL .

- **MOVE A TO D**

تنسخ محتويات العنصر A فى العنصر D ، حيث تملأ صورة العنصر D هي PIC X(7) المواقع الأكثر بفراغات من ناحية اليمين : بعد تنفيذ عبارة النقل.. يحتوى A على HELLO، بينما يحتوى D على HELLObb .

- **MOVE A TO B C D**

تنسخ محتويات عنصر البيانات A فى عناصر البيانات : B و C و D وينتج عنها نفس لغة الآلة مثل عبارات النقل الثالث السابقة تماماً .

- **MOVE SPACES TO A**

تنسخ خمس فراغات (هنا literal هو الثابت الاستعارى SPACES) فى عنصر البيانات A (identifier-2)، والمعروف بصورة PIC X(5) .

- **MOVE ZEROS TO A C**

تضع 00000 فى A، وتضع 000 فى C .

- **MOVE ALL "-" TO C D**

تضع --- فى C وتضع ----- فى D .

لاحظ مرة أخرى.. استخدام الثابت الاستعارى " - ALL " ، الذى يقدم العدد الصحيح والنوع الصحيح للرموز تلقائياً

مثال ٦ - ٢٦ :

```
01 SAMPLE-NUMERIC-ITEMS.
05 A PIC S9(5) VALUE +12345.
05 B PIC S9(5).
05 C PIC S9(3).
05 D PIC S9(7).
```

- **MOVE A TO B**

حيث إن كلا من A و B له صورة PIC 9(5)، ويسبب استخدام DISPLAY ، فإنهما يحتويان نفس القيمة بعد تنفيذ عبارة النقل : +12345 .

- MOVE A TO C

يحدث الحذف من ناحية اليسار بالنسبة الى العناصر العديدة، يحتوى C على +345 [حيث أن صورته هي (3) S9] ، ويظل A كما هو دون تغيير .

- MOVE A TO D

تضاف الأصفار الزائدة للعناصر العديدة من ناحية اليسار، ويظل A كما هو دون تغيير ، بينما يحتوى D على +0012345 .

مثال ٦ - ٢٧ :

بمعرفة مايلي :

01 SAMPLE-NUMERIC-ITEMS.

05	A	PIC S9(3)V99	COMP-3	VALUE +123.45.
05	B	PIC S9(3)V99	DISPLAY.	
05	C	PIC S9(3)V99	COMP.	
05	D	PIC S9(5)V99	COMP-3.	
05	E	PIC S9(5)V9	DISPLAY.	
05	F	PIC S9(5)V999	COMP.	
05	G	PIC S9(2)V9	COMP-3.	

- MOVE A TO B

A و B لهما نفس الصورة ، ولهما استخدام USAGE مختلف . وتحول قيمة COMP-3 في A الى DISPLAY ، وتنسخ في B . وعلى هذا ، يحتوى كل من A و B على +123.45 بعد النقل ، إلا أن قيمة A تخزن في صورة COMP-3 ، أما قيمة B فتخزن في صورة DISPLAY .

- MOVE A TO D

الاستخدام هنا هو نفسه ، ولكن الصورة مختلفة . تضبط العلامات العشرية ، ثم توضع أصفار إضافية في الحقل المستقبل، وذلك من ناحية اليسار . يظل A كما هو دون تغيير ، ويحتوى D على +00123.45 في صورة COMP-3 . تحفظ العلامات العشرية في الحقول الراسلة والمستقبلة مضبوطة دائماً .

● MOVE A TO E

يحدث تحويل تلقائي من COMP-3 الى DISPLAY . بالإضافة إلى ذلك ، تضاف أصفار إلى الجزء الصحيح من العدد (من ناحية اليسار) .

وحيث إن E لها موقع لكسر عشري واحد ، E تحتوى على 00123.4+ والاستخدام DISPLAY بعد النقل . لاحظ أنه لم يحدث تقريب rounding بواسطة عبارة النقل ، وأن العلامة العشرية ظلت في نفس موقعها .

● MOVE A TO F

يحدث هنا تحويل تلقائي من COMP-3 الى COMP . أضيفت أصفار إلى الرقم الصحيح من العدد (من ناحية اليسار). حيث أن الحقل المستقبل يحدد مواقع عشرية أكثر من الحقل الراسل.. تضاف أصفار كذلك في الكسر العشري (من ناحية اليمين) . يظل A كما هو دون تغيير بعد النقل ويحتوى F على 00123.450+ والاستخدام COMP .

● MOVE A TO G

كلا الاستخدامين من نوع COMP-3 ، بحيث أنه لاينفذ أى تحويل، ونظراً لأن صورة G هي PIC S 9(2)V9 ، يحدث حذف من ناحية اليمين وناحية اليسار . وتكون نتيجة G هي 23.4+ من نوع COMP-3 .

● MOVE A TO B D E F G

ينتج عن هذه العبارة نفس لغة الآلة تماماً، مثل العبارات الخمس السابق ذكرها .

● MOVE ZEROS TO A B C

حيث إن كلاً من A , B , C له نفس الصورة.. فإن كلا منها يحتوى على 000.00+ . مع ملاحظة أن كل حقل يحتوى على نوع صفر ZERO مختلف، كمايلي: A يكون من نوع COMP-3 , B من نوع DISPLAY , C من نوع COMP . وعندما تستخدم الثوابت الاستعارية في عبارة النقل أو في جزء VALUE ، فإنها تمثل الكمية الصحيحة correct amount من البيانات، والاستخدام USAGE الصحيح دائماً .

مثال ٦ - ٢٨ :

بمعرفة مايلي :

```
01 SAMPLE-NUMERIC-ITEMS.
05 A          PIC S9(5)V99      VALUE +00010.08.
05 A-EDITED   PIC $$$,$$$.$99-.
MOVE A TO A-EDITED
```

عندما ينقل عنصر عددي إلى حقل به رموز تنقيح.. يشمل جزء من تنفيذ عبارة النقل تنقيحاً للقيمة في الحقل المستقبل .
وتظل القيمة في A كما هي دون تغيير ، أما القيمة في A-EDITED بعد النقل فهي : bbbb\$10.058b

مثال ٦ - ٢٩ :

(عبارات نقل غير صحيحة) : يصنف الكويز عناصر البيانات في ست فئات بيانات categories of data وباعتبار lit eral-2 و eral-1 فئتين من هذه الفئات ، تكون لدينا القائمة التالية بعدد ١٤ عبارة نقل صحيحة .

- MOVE ALPHABETIC-DATA TO NUMERIC-INTEGGER-DATA
- MOVE ALPHABETIC-DATA TO NUMERIC-NONINTEGGER-DATA
- MOVE ALPHABETIC-DATA TO NUMERIC-EDITED-DATA
- MOVE ALPHANUMERIC-EDITED-DATA TO NUMERIC-INTEGGER-DATA
- MOVE ALPHANUMERIC-EDITED-DATA TO NUMERIC-NONINTEGGER-DATA
- MOVE ALPHANUMERIC-EDITED-DATA TO NUMERIC-EDITED-DATA
- MOVE NUMERIC-INTEGGER-DATA TO ALPHABETIC-DATA
- MOVE NUMERIC-NONINTEGGER-DATA TO ALPHABETIC-DATA
- MOVE NUMERIC-NONINTEGGER-DATA TO ALPHANUMERIC-DATA
- MOVE NUMERIC-NONINTEGGER-DATA TO ALPHANUMERIC-EDITED-DATA
- MOVE NUMERIC-EDITED-DATA TO ALPHABETIC-DATA
- MOVE NUMERIC-EDITED-DATA TO NUMERIC-INTEGGER-DATA
- MOVE NUMERIC-EDITED-DATA TO NUMERIC-NONINTEGGER-DATA
- MOVE NUMERIC-EDITED-DATA TO NUMERIC-EDITED-DATA

لاحظ أنه لا يسمح بالخليط، عندما يكون من عناصر بيانات حرفية أو حرفية عددية، أو منقحة مع عناصر بيانات عددية.
يسمح بأي خليط آخر غير مذكور أعلاه .

مثال ٦ - ٣٠ :

كانت كل عبارات النقل الموضحة حتى الآن من عنصر بيانات فردي لعنصر آخر . اعتبر محاولة نقل المجموعة التالية :

01	INPUT-INVOICE-RECORD.		
05	INPUT-VENDOR-ID	PIC X(5).	
05	INPUT-INVOICE-ID	PIC X(4).	
05	INPUT-NUMBER-ITEMS	PIC S9(4).	
05	INPUT-AMOUNT	PIC S9(5)V99.	
01	PAYABLE-INVOICE-RECORD.		
05	PAYABLE-VENDOR-ID	PIC X(5).	
05	PAYABLE-INVOICE-ID	PIC X(4).	
05	PAYABLE-NUMBER-ITEMS	PIC S9(4)	COMP-3.
05	PAYABLE-AMOUNT	PIC S9(5)V99	COMP-3.

MOVE INPUT-INVOICE-RECORD TO PAYABLE-INVOICE-RECORD

عند نقل مجموعة عناصر (على أى مستوى) ، أو عند استقبال مجموعة عناصر لقيمة منقولة .. فإنها تعامل كعنصر حرفى
عددى كبير .

هنا .. لكل الحقول فى INPUT- INVOICE- RECORD استخدام DISPLAY ، ويبلغ طول مجموعة العناصر 20
بايت (5 + 4 + 4 + 7) ونقلت كما لو كانت معرفة بصورة PIC X(20) . يبلغ طول PAYABLE- INVOICE- RECORD
16 بايت (5 + 4 + 3 + 4) . لاحظ الفرق الذى يحدث بسبب COMP-3 ، المستخدمة مع العناصر العددية) ، وعلى هذا -
ولأغراض النقل - فإنه يعامل كما لو كان معرفا بالصورة PIC X(16) . أما تنفيذ عبارة النقل .. فإنه .

(١) لايحول الحقول الجزئية INPUT- NUMBER- ITEMS , INPUT- AMOUNT من DISPLAY إلى COMP-3 ,
قبل نقلها على التوالى إلى PAYABLE- AMOUNT , PAYABLE- NUMBER - ITEMS .

(٢) يحذف الأربع بايت الموجودة فى أقصى يمين INPUT- INVOICE- RECORD [الفرق بين PIC X(16) ,
(20)] . النتيجة هى نفايا موجودة فى PAYABLE - NUMBER - ITEMS , PAYABLE- AMOUNT التى
تتسبب فى نهاية البرنامج نهاية غير طبيعية إذا ما أشير لهذه الحقول .

مثال ٦ - ٢١ :

نستدل من مثال ٦ - ٣٠ على أن نقل مجموعة غير آمن إلا إذا حدث وكان :

(١) لكل الحقول الجزئية فى كل من العنصر الراسل والمستقبل استخدام DISPLAY ، كما أنه يتوقع حدوث أى تأثير نتيجة
الاختلافات فى الأطوال ، أو فى ترتيب الحقول الجزئية من قبل المبرمج .

أو ، أيضا

(٢) لاتعد كل الحقول الجزئية من نوع DISPLAY ، ولكن العنصر الراسل والعنصر المستقبل يناظران بعضهما البعض تماما
من ناحية مواقع الحقول الجزئية كلها والصورة والاستخدام

اعتبر مجموعتى العناصر التاليتين :

01	PENDING-INVOICE-RECORD.	
05	PENDING-INVOICE-ID	PIC X(4).
05	PENDING-VENDOR-ID	PIC X(5).
05	PENDING-AMOUNT-DUE	PIC S9(5)V99.
05	PENDING-NUMBER-ITEMS	PIC S9(4).
01	PAYABLE-INVOICE-RECORD.	
05	PAYABLE-VENDOR-ID	PIC X(5).
05	PAYABLE-AMOUNT-DUE	PIC S9(5)V99.
05	PAYABLE-INVOICE-ID	PIC X(4).
05	PAYABLE-NUMBER-ITEMS	PIC S9(4).

حيث إن الشرط الأول سالف الذكر لم يتحقق... فإن :

MOVE PENDING-INVOICE-RECORD TO PAYABLE-INVOICE-RECORD

لاتعمل . بسبب الترتيب المختلف للحقول الجزئية في مجموعتي العناصر.. ينقل PENDING- INVOICE- ID (4 بايت)، وأول بايت من PENDING- INVOICE- ID الى PAYABLE- INVOICE- ID (5 بايت) ... الخ . وتنتج عن هذا بالطبع- أخطاء عندما يشار الى حقول PAYABLE- INVOICE- RECORD . الطريقة الصحيحة لتحقيق النتائج المرجوة، هي نقل الحقول الجزئية حقلا حقلا :

```
MOVE PENDING-INVOICE-ID TO PAYABLE-INVOICE-ID
MOVE PENDING-VENDOR-ID TO PAYABLE-VENDOR-ID
MOVE PENDING-AMOUNT-DUE TO PAYABLE-AMOUNT-DUE
MOVE PENDING-NUMBER-ITEMS TO PAYABLE-NUMBER-ITEMS
```

(في بعض المؤسسات.. يمكن أن يتأثر نقل مجموعات معينة بعبارة نقل على النحو التالي :

2- MOVE CORRESPONDING identifier-1 to identifier-2 ، إلا أننا نوصى - بشدة - باستخدام نقل العناصر الفردية بدلا من ذلك) .

٦ - ٩ إيقاف تنفيذ البرنامج :

عبارة التوقف

إيقاف التنفيذ نهائيا

تتسبب عبارة التنفيذ STOP RUN في أن يعيد برنامج الهدف تحكم الكمبيوتر الى نظام التشغيل . ويفصل هذا الإجراء هذا التنفيذ للبرنامج تماما عن الكمبيوتر . ويجب أن تغلق كل الملفات قبل تنفيذ STOP RUN وإلا فإنه يمكن أن تحدث أخطاء جسيمة أثناء تشغيل لاحق للملف . يجب أن توجد في كل برنامج عبارة STOP RUN واحدة على الأقل ، يتطلب عدد من نمطيات الكتابة وجود عبارة STOP RUN واحدة فقط .

إيقاف التنفيذ لحظيا

صيغة إيقاف التنفيذ لحظيا هي STOP literal ،وتؤدى مايلى :

(١) عرض قيمة الثابت literal عن طريق النهاية الطرفية التى يستخدمها مشغل الكمبيوتر.

(٢) إيقاف تنفيذ برنامج الهدف لحظيا (بينما يقوم المشغل بتغيير شرائط أو صيغ ورق الطابع، أو غيرها) حتى يعطى مشغل الكمبيوتر إشارة لنظام التشغيل عن طريق نهايته الطرفية ، وذلك بالاستمرار فى تنفيذ برنامج الهدف .

مثال ٦ - ٣٢ :

PROCEDURE DIVISION.

OPEN INPUT PAYABLE-INVOICE-FILE

STOP "PLEASE MOUNT PREPRINTED CHECK FORMS IN PRINTER"

OPEN OUTPUT PAID-INVOICE-CHECK-FILE

بعد قراءة الرسالة... PLEASE MOUNT... على النهاية الطرفية.. يفترض أن يكون مشغل الكمبيوتر قد أعد الطابع لطباعة شيكات ، وعند ذلك يعيد بدء البرنامج عند ... OPEN OUTPUT .

٦ - ١٠ تنفيذ متحكم فيه لمقطع :

عبارة التنفيذ

تسمح عبارة التنفيذ PERFORM بتنفيذ محتويات مقطع مرة واحدة أو أكثر من أى نقطة فى البرنامج تختلف عن الترتيب التابعى الذى تنفذ فيه عبارات البرنامج . والصيغة البسيطة لعبارة التنفيذ، هى كما يلى :

PERFORM paragraph-name

حيث إن paragraph- name هو اسم مقطع يعرفه المبرمج فى جزء الإجراءات .. فأن تنفيذ هذه الصيغة يتسبب فى تنفيذ كل العبارات الموجودة فى المقطع المحدد، فى ترتيب طبيعى (الأولى فالثانية وهكذا) . يستمر التنفيذ المتتالى الطبيعى للتعليمات بعد ذلك بالعبارات التى تلي عبارة PERFORM مباشرة .

مثال ٦ - ٣٣ :

PROCEDURE DIVISION.

OPEN INPUT TIME-CARD-FILE
OUTPUT TIME-CARD-LISTING

PERFORM INITIALIZE-COUNTERS-SWITCHES

WRITE LISTING-LINE
FROM WS-LISTING-LINE-HEADING
AFTER ADVANCING PAGE

```

PERFORM INPUT-TIME-RECORD

MOVE WS-TIME-CARD-ID TO WS-LISTING-ID
.....
INITIALIZE-COUNTERS-SWITCHES.

MOVE "F" TO TIME-CARD-END-SW
MOVE ZEROS TO WS-EMPLOYEE-COUNTER
                WS-AMOUNT-OVERTIME
                WS-TOTAL-HOURS
INPUT-TIME-RECORD.

READ TIME-CARD-FILE
  INTO WS-TIME-CARD-RECORD
  AT END
    MOVE "T" TO TIME-CARD-END-SW

```

يوجد عبارتين PERFORM فإن ترتيب التنفيذ السابق يكون كما يلي :

PERFORM INITIALIZE... (٢)	OPEN... (١)
MOVE ZEROS... (٤)	MOVE " F "... (٣)
PERFORM INPUT-... (٦)	WRITE... (٥)
MOVE WS_... (٨)	READ (٧)

تحتاج معظم برامج الأعمال إلى تكرار دورة المدخلات والتشغيل والمخرجات لكل سجل منطقي من سجلات ملف المدخلات . هذا يعني أن منطق البرنامج program logic يجب أن يحتوى على دورة loop ؛ حيث تنفذ مجموعة من التعليمات مرات ومرات . وعبرة الكويل الأكثر فائدة في دورات البرمجة لها التكوين التالي:

PERFORM paragraph-name UNTIL condition

عندما تنفذ مثل هذه العبارة.. ينفذ المقطع المحدد مرات ومرات حتى يتحقق الشرط المحدد . وتناقش الشروط بالكامل في الفصل السابع من الكتاب ، والشرط حاليا له الشكل التالي data- name EQUAL literal .

مثال ٦ - ٢٤ :

PROCEDURE DIVISION.

```

OPEN   INPUT   TIME-CARD-FILE
        OUTPUT  TIME-CARD-LISTING

MOVE "F" TO TIME-CARD-END-SW

PERFORM PRODUCE-TIME-LISTING-LINE
  UNTIL TIME-CARD-END-SW EQUAL "T"

```

CLOSE TIME-CARD-FILE
TIME-CARD-LISTING

STOP RUN

PRODUCE-TIME-LISTING-LINE.

READ TIME-CARD-FILE
INTO WS-TIME-RECORD
AT END
MOVE "T" TO TIME-CARD-END-SW

IF TIME-CARD-END-SW EQUAL "F"

MOVE WS-TIME-REC-ID TO WS-LISTING-ID
MOVE WS-TIME-REC-NAME TO WS-LISTING-NAME
MOVE WS-TIME-REC-HOURS TO WS-LISTING-HOURS

WRITE TIME-CARD-LISTING-LINE
FROM WS-LISTING-LINE
AFTER ADVANCING 2 LINES

دعنا نفترض أنه لا يوجد سوى سجلين منطقيين فقط في الملف TIME-CARD-FILE. والترتيب المنطقي الذي تنفذ به

العبارات، هو مايلي :

OPEN...	(١)	
MOVE "F" TO TIME-CARD-END-SW	(٢)	
PERFORM PRODUCE-TIME-LISTING-LINE UNTIL...	(٢)	
(حيث TIME-CARD-END-SW ليس صحيحا (T) ، فينفذ المقطع لأول مرة)		
READ TIME-CARD-FILE... AT END MOVE "T" TO TIME-CARD-END-SW	(٤)	<div style="display: flex; align-items: center;"> <div style="font-size: 4em; margin-right: 10px;">}</div> <div> <p>اول</p> <p>تنفيذ</p> <p>للمقطع</p> </div> </div>
(إدخال أول سجل منطقي.. يظل المفتاح "F")		
IF TIME-CARD-END-SW EQUAL "F"	(٥)	
(حيث أن الشرط صحيح.. فتتخذ كل العبارات حتى النقطة)		
MOVE WS-TIME-REC-ID...	(٦)	
MOVE WS-TIME-REC-NAME...	(٧)	
MOVE WS-TIME-REC-HOURS...	(٨)	
WRITE TIME-CARD-LISTING-LINE FROM...	(٩)	
(بعد آخر عبارة في المقطع.. يتم تقويم الشرط. وحيث أن المفتاح ليس "T" يعاد تنفيذ المقطع مرة ثانية)		
READ TIME-CARD-FILE... AT END MOVE "T" TO TIME-CARD-END-SW	(١٠)	<div style="display: flex; align-items: center;"> <div style="font-size: 4em; margin-right: 10px;">}</div> <div> <p>التنفيذ</p> <p>الثاني</p> <p>للمقطع</p> </div> </div>
(إدخال ثاني منطقي ، يظل المفتاح "F")		
IF TIME-CARD-END-SW EQUAL "F"	(١١)	
(حيث إن الشرط صحيح، تنفذ كل العبارات حتى النقطة)		
MOVE WS-TIME-REC-ID...	(١٢)	
MOVE WS-TIME-REC-NAME...	(١٣)	
MOVE WS-TIME-REC-HOURS...	(١٤)	
WRITE TIME-CARD-LISTING-LINE FROM...	(١٥)	
(بعد آخر عبارة في المقطع.. يتم تقويم الشرط. وحيث إن المفتاح ليس "T" يعاد تنفيذ المقطع مرة ثانية)		
READ TIME-CARD-FILE... AT END MOVE "T" TO TIME-CARD-END-SW	(١٦)	

- التنفيذ الثالث للمقطع] عند READ الثالثة الملف الموجود فيه سجلان منطقيان فقط.. تحدث نهاية الملف .. ينفذ AT END...]
- جاءلا المفتاح ، محتويا على "T" ، وتستمر PERFORM حتى نهاية المقطع) .
- IF TIME-CARD-END-SW EQUAL "F" (١٧)
- (المفتاح الآن "T" ، وتترك كل العبارات حتى النقطة التي تنتهى IF... هذه النقطة تنتهى المقطع كذلك ،
جاءلة IF... هي آخر عبارة تنفذ فى البرنامج) .
- CLOSE... (١٨)
- (بعد تنفيذ آخر عبارة فى مقطع PERFORM .. يختبر الكمبيوتر المفتاح . وحيث إنه يحتوى على "T"
الآن ، فينتهى تنفيذ عبارة PERFORM وتنفذ العبارة التالية CLOSE...) .
- STOP RUN (١٩)
- (بعد تنفيذ STOP RUN .. لاتنفذ أى عبارة أخرى من عبارات البرنامج) .
- لاحظ كيف يتجنب منطق البرنامج - بعناية تشغيل سجل بعد حدوث نهاية الملف (ارجع الى مثال ٦ - ١٠) .

مثال ٦ - ٢٥ :

- بعد توضيح عمل الدورات.. نكون الآن فى موقع يسمح لنا بفحص برنامج كويل كامل .. اعتبر تطبيق الرواتب الذى يملأ فيه العاملون استمارات وقت، طبقا لأرقام تعريفهم وأسمائهم، وإجمالى عدد ساعات عملهم الأسبوعية . يوقع الملاحظ استمارات الوقت هذه، ويرسلها إلى تشغيل البيانات ؛ حيث يتم إدخال المعلومات باستخدام معدات من لوحة المفاتيح إلى القرص مباشرة key- to- disk . والمرغوب فيه وجود برنامج تنقيح edit program ، يطبع نسخة دائمة من سجلات الوقت الأسبوعية لفحصها بصريا وتصحيحها .
- قائمة مصدر مثل هذا البرنامج مدونة أدناه . ونأمل أن يكون القارئ قادرا على تعريف التطبيقات لأساسيات برمجة الكويل التى سبق مناقشتها حتى الآن . وبصفة خاصة :
- (١) أجزاء ORGANIZATION و ACCESS فى عبارات SELECT ، المسهب فيها (حيث ان الحالة التقليدية هى SE QUENTIAL) ، والتى تحدد لتوثيق البرنامج .
- (٢) استخدام أجهزة كمبيوتر IBM لجزء 0 BLOCK CONTAINS ملفات مدخلات أقراص .
- (٣) حذف جزء BLOCK CONTAINS من ملف الطباعة .
- (٤) استخدام LABEL RECORDS ARE STANDARD لملف القرص ، واستخدام OMITTED لملف الطابع .
- (٥) إجراء كل تشغيل السجل فعلا فى مخزن العمل باستخدام READ... INTO و WRITE... FROM ، ويكون وصف السجل فى قسم الملفات مثل PIC X (80) و PI X(132) فقط ، حيث يحدد التخطيط الفعلى للسجل فى نسخة مخزن العمل .
- (٦) تجميع عناصر مخزن العمل تحت فئات على المستوى 01 ، وسرد العناصر بطريقة منظمة .
- (٧) استخدام جزء VALUE فى عمل كل من أسطر العناوين وأسطر أرقام التعريف، والاسم، والساعات فى مخزن العمل .
- (٨) استخدام FILLER لكل عناصر البيانات التى لا يشار اليها فى جزء الإجراءات .
- (٩) يستخدم جزء الإجراءات عبارات PERFORM UNTIL... ، PERFORM استخداما واسعا للتحكم فى سريان منطق البرنامج وعمل الدورات الأساسية للمدخلات والتشغيل والمخرجات .
- (١٠) أقناع نفسك بأن هذا البرنامج لايقوم بتشغيل سجل به نفايا بعد الوصول إلى نهاية الملف (وأنه يقوم بتشغيل كل السجلات المنطقية فى ملف القرص المحتوى على معلومات استمارات الوقت) .
- (١١) ملاحظة الترحيل فى جزء الإجراءات ، والترحيل وضبط الأعمدة فى جزء البيانات .

00001 IDENTIFICATION DIVISION.

00003 PROGRAM-ID. HRVERIFY.

00004

00005 AUTHOR. LARRY NEWCOMER.

00006 INSTALLATION. PENN STATE UNIVERSITY, YORK CAMPUS.

00007

00008 DATE-WRITTEN. MAY 1983.

00009 DATE-COMPILED. MAY 9,1983.

00011 SECURITY. NONE.

00012

00013 * HRVERIFY PRODUCES A PRINTED LISTING OF THE WEEKLY PAYROLL

00014 * TIME FILE (PREPARED BY KEY-TO-DISK OPERATORS USING WEEKLY

00015 * TIME SHEETS). THIS LISTING IS USED TO VISUALLY VERIFY

00016 * THE CORRECTNESS OF EMPLOYEE ID NUMBERS, EMPLOYEE NAMES,

00017 * AND EMPLOYEE WEEKLY HOURS WORKED AS TYPED BY DATA ENTRY.

00018 * THE ONLY OUTPUT IS THIS VISUAL EDIT REPORT.

00020 ENVIRONMENT DIVISION.

00022 CONFIGURATION SECTION.

00024 SOURCE-COMPUTER. IBM-3081.

00025 OBJECT-COMPUTER. IBM-3081.

00027 INPUT-OUTPUT SECTION.

00028

00029 FILE-CONTROL.

00031 SELECT TIME-SHEET-INPUT-FILE

00032 ASSIGN TO TIMECARD

00033 ORGANIZATION IS SEQUENTIAL

00034 ACCESS IS SEQUENTIAL

00035 .

00036

00037 SELECT VERIFY-TIME-REPORT

00038 ASSIGN TO TIMELIST

00039 ORGANIZATION IS SEQUENTIAL

00040 ACCESS IS SEQUENTIAL

00041 .

00043 DATA DIVISION.

00045 FILE SECTION.

00047 FD TIME-SHEET-INPUT-FILE

00048 BLOCK CONTAINS 0 RECORDS

00049 RECORD CONTAINS 80 CHARACTERS

00050 LABEL RECORDS ARE STANDARD

00051 .

00052

00053 01 TIME-SHEET-INPUT-RECORD PIC X(80).

00055 FD VERIFY-TIME-REPORT

00056 RECORD CONTAINS 132 CHARACTERS

00057 LABEL RECORDS ARE OMITTED

00058 LINAGE IS 50

00059 WITH FOOTING AT 45

00060 LINES AT TOP 8

00061 LINES AT BOTTOM 8

00062 .

00063

00064 01 VERIFY-TIME-LINE PIC X(132).


```

00066      WORKING-STORAGE SECTION.

00068      01 PROGRAM-FLAGS-AND-SWITCHES.
00069          05 WS-TIME-SHEET-END-SW      PIC X.

00071      01 WORKING-INPUT-RECORD-AREAS.

00073          05 WS-TIME-SHEET-REC.
00074              10 WS-TIME-SHEET-ID      PIC X(4).
00075              10 WS-TIME-SHEET-NAME    PIC X(20).
00076              10 WS-TIME-SHEET-HOURS  PIC S9(2)V9.
00077              10 FILLER                PIC X(53).

00079      01 WORKING-OUTPUT-RECORD-AREAS.

00081          05 WS-VERIFY-HEADING.
00082              10 FILLER                PIC X(4)      VALUE "-ID-".
00083              10 FILLER                PIC X(4)      VALUE SPACES.
00084              10 FILLER                PIC X(20)     VALUE "NAME".
00085              10 FILLER                PIC X(4)      VALUE SPACES.
00086              10 FILLER                PIC X(5)      VALUE "HOURS".
00087              10 FILLER                PIC X(95)     VALUE SPACES.

00089          05 WS-VERIFY-LINE.
00090              10 WS-VERIFY-ID          PIC X(4).
00091              10 FILLER                PIC X(4)      VALUE SPACES.
00092              10 WS-VERIFY-NAME       PIC X(20).
00093              10 FILLER                PIC X(4)      VALUE SPACES.
00094              10 WS-VERIFY-HOURS     PIC Z9.9-.
00095              10 FILLER                PIC X(95)     VALUE SPACES.

00097      PROCEDURE DIVISION.

00099          PERFORM INITIALIZE-FILES-AND-SWITCHES
00100
00101          PERFORM PRODUCE-VERIFY-LINE
00102              UNTIL WS-TIME-SHEET-END-SW EQUAL "T"
00103
00104          CLOSE    TIME-SHEET-INPUT-FILE
00105                  VERIFY-TIME-REPORT
00106
00107          STOP RUN
00108
00110      INITIALIZE-FILES-AND-SWITCHES.

00112          MOVE "F" TO WS-TIME-SHEET-END-SW
00113          OPEN    INPUT  TIME-SHEET-INPUT-FILE
00114                  OUTPUT VERIFY-TIME-REPORT
00115          PERFORM READ-TIME-SHEET-FILE
00116          WRITE VERIFY-TIME-LINE
00117              FROM WS-VERIFY-HEADING
00118                  AFTER ADVANCING PAGE
00119
00121      READ-TIME-SHEET-FILE.

00123          READ TIME-SHEET-INPUT-FILE
00124              INTO WS-TIME-SHEET-REC
00125              AT END
00126                  MOVE "T" TO WS-TIME-SHEET-END-SW
00127

```

```

00129          PRODUCE-VERIFY-LINE.

00131          MOVE WS-TIME-SHEET-ID          TO WS-VERIFY-ID
00132          MOVE WS-TIME-SHEET-NAME        TO WS-VERIFY-NAME
00133          MOVE WS-TIME-SHEET-HOURS       TO WS-VERIFY-HOURS
00134          WRITE VERIFY-TIME-LINE
00135          FROM WS-VERIFY-LINE
00136          AFTER ADVANCING 3 LINES
00137          AT END-OF-PAGE
00138          WRITE VERIFY-TIME-LINE
00139          FROM WS-VERIFY-HEADING
00140          AFTER ADVANCING PAGE
00141          .
00142          PERFORM READ-TIME-SHEET-FILE
00143          .
    
```

ويمكن أن تشبه المخرجات من هذا البرنامج مايلي :

0027 FOLKERS DEIRDRE 45.6

٦ - ١١ تشغيل : عبارة الجمع

تتسبب عبارة الجمع ADD في جمع محتويات عنصرين بيانات (أو أكثر) من النوع العددي، ووضع حاصل الجمع في واحد (أو أكثر) من عناصر البيانات هذه . وهناك صيغتان رئيسيتان لهذه العبارة ، وهما : ADD... و ADD... GIVING... ، TO ... (ولا توصى بصيغة ثالثة ADD CORRESPONDING مع حذفها من هذا الكتاب) .

اضف ... إلى

هذا البديل ... TO ... ADD ، له التكوين المبين في شكل (٦ - ٤) .

$$\text{ADD } \left\{ \begin{array}{l} \text{identifier-1} \\ \text{literal-1} \end{array} \right\} \left\{ \begin{array}{l} \text{identifier-2} \\ \text{literal-2} \end{array} \right\} \dots$$

$$\text{TO identification-m [ROUNDED][identification-n [ROUNDED]]}$$

$$[\text{ON SIZE ERROR imperative-statement}]$$

شكل (٦ - ٤)

مثال (٦ - ٣٦) :

- ADD WEEKLY-HOURS TO TOTAL-HOURS

تضاف المحتويات الحالية للعنصر WEEKLY- HOURS الى المحتويات الحالية للعنصر TOTAL- HOURS ويوضع حاصل الجمع في TOTAL- HOURS ، ولاتتغير محتويات WEEKLY- HOURS . يجب أن يكون كل من العنصرين عدديا . (وليس منقحا) . وينتج المترجم - بصورة تلقائية - لغة آلة إضافية لمعالجة ضبط العلامات العشرية، وتحويل الصورة من استخدام USAGE معين لاستخدام آخر .

- ADD ITEM-A ITEM-B TO ITEM-C

تضاف محتويات العناصر ITEM- A , ITEM- B , ITEM- C مع بعضها ، ويوضع حاصل الجمع في العنصر ITEM- C ، وتظل كل من محتويات ITEM- A , ITEM- B كما هي دون تغيير. يجب أن تكون كل عناصر البيانات عددية (وايست منقحة).

- ADD ITEM-A ITEM-B ITEM-C TO ITEM-D ITEM-E

يوضع مجموع العناصر ITEM- A , ITEM- B , ITEM- C , ITEM- D في العنصر ITEM- D ، ومجموع العناصر ITEM- A , ITEM- B , ITEM- C و ITEM- E يوضع في ITEM- E . تظل قيم ITEM- B , ITEM- C , ITEM- A كما هي دون تغيير .

- ADD 1 TO NUMBER-OVERTIME-EMPLOYEES

يضاف 1 الى المحتويات الحالية للعنصر NUMBER- OVERTIME- EMPLOYEES (والذي يتزايد على ذلك بمقدار (1) . استخدام الثوابت العددية معتاد في الحسابات .

اضف ... معطيا

لهذا البديل ADD... GIVING... التكوين المبين في شكل (٦ - ٥) .

```
ADD {identifier-1} {identifier-2} {identifier-3} ...
    {literal-1} {literal-2} {literal-3} ...
    GIVING identifier-m [ROUNDED] [identifier-n [ROUNDED]] ...
    [ON SIZE ERROR imperative-statement]
```

شكل (٦ - ٥)

مثال ٦ - ٢٧ :

- ADD REGULAR-HOURS OVERTIME-HOURS GIVING TOTAL-HOURS

يتطلب بديل GIVING عنصرى بيانات على الأقل ، يسار كلمة GIVING وعنصر بيانات واحد على الأقل على يمينها . تضاف المحتويات الفعلية لكل من REGULAR- HOURS و OVERTIME- HOURS ، ويوضع حاصل الجمع فى TO- HOURS . وتظل محتويات REGULAR- HOURS و OVERTIME- HOURS كما هى بون تغيير . ولا تدخل المحتويات الاصلية للعنصر TOTAL - HOURS فى عملية الجمع . يجب أن تكون عناصر البيانات الموجودة على يسار GIV- ING ، والتي تجمع مع بعضها البعض ، عددية وغير منقحة . تستقبل عناصر البيانات الموجودة على يمين GIVING نتيجة الجمع ولا تجمع هى بنفسها ، وعلى هذا .. فبالرغم من أنها يجب أن تكون عددية إلا أنها يمكن أن تكون منقحة .

- ADD PAYABLE-INVOICE-AMOUNT
PAYABLE-INVOICE-FREIGHT
PAYABLE-INVOICE-SPECIAL- GIVING OUTPUT-CHECK-AMOUNT

تسهل طريقة الكتابة من القراءة ومن إجراء التعديلات . يجب أن تكون كل حقول PAYABLE- INVOICE عددية وغير منقحة . تجمع الحقول ويوضع حاصل جمعها بدلا من محتويات OUTPUT- CHECK- AMOUNT (الذى لا يدخل فى الجمع) ، ويكون الحقل OUTPUT- CHECK- AMOUNT منقحا للطباعة .

- ADD STATE-TAX
FEDERAL-TAX GIVING PAYROLL-REPORT-TOTAL-TAX
EMPLOYEE-MASTER-TOTAL-TAX

يجب أن تكون كل الحقول عددية ، إلا PAYROLL- REPOT- TOTAL- TAX يكون منقحا (يسرى هذا عند استقبال نتائج) . يجمع STATE- TAX و FEDERAL- TAX ويوضع حاصل جمعهما فى PAYROLL- REPORT- TOTAL- TAX وفى EMPLOYEE- MASTER- TOTAL- TAX

مثال ٦ - ٢٨ :

- ADD STATE-TAX TO FEDERAL-TAX GIVING TOTAL-TAX

خطأ تكوينى : لا يمكن استخدام TO ، و GIVING معا .

- ADD FREIGHT-CHARGES TO INVOICE-AMOUNT AND SHIPPING-TOTAL

AND غير صحيحة فى كل من صيغتي

- ADD DAILY-HOURS GIVING WEEKLY-HOURS MONTHLY-HOURS

تتطلب صيغة GIVING قيمتين على الأقل، على يسار كلمة GIVING . وربما كان المقصود مايلي :

ADD DAILY-HOURS TO WEEKLY-HOURS MONTHLY-HOURS

يوضح استخدام جزء ROUNDED وجزء ON SIZE ERROR مع كل من الصيغتين للجمع ADD فى القسمين 12 و 13 - الفصل السادس .

٦ - ١٢ تشغيل : حذف جزء من الكسر العشري

و جزء التقريب

كل العبارات الحسابية (ADD , SUBTRACT , MULTIPLY , DIVIDE , COMPUTE) تنتج تلقائيا تعليمات بلغة الآلة.

(١) لضبط العلامات العشرية للقيم التى تجمع أو تطرح

(٢) التحويل من أحد الاستخدامات USAGE إلى استخدام آخر والعكس .

من الممكن أن تكون نتيجة الحسابات لها مواقع للكسر العشري، أكثر من المواقع المحددة له فى الحقل المستقبل . والإجراء التقليدى ، فى هذه الحالة هو حذف العدد اللازم من الأرقام من ناحية اليمين .

لتعديل عملية الحذف ، يمكن وضع كلمة التقريب ROUNDED بعد اسم أى حقل يستقبل النتيجة العملية الحسابية (راجع الشكلين رقم ٦ - ٤ ، ٦ - ٥) . مع تحديد ROUNDED :

(١) إذا كان الرقم الموجود على أقصى اليسار من الجزء المحذوف ٥ أو أكثر.. يضاف 1 إلى الرقم المجاور له، والموجود فى اليمين (أى يحدث تقريب) للأرقام المتبقية .

(٢) إذا كان الرقم الموجود على أقصى اليسار من الجزء المحذوف ٤ أو أقل ، فتحذف النتيجة بيساطة ، دون تغيير للرقم الموجود فى أقصى يمين الأرقام المتبقية .

مثال ٦ - ٢٩ :

```
05 ANSWER PIC S99V9
.....
ADD X Y Z GIVING ANSWER
```

إذا كان مجموع X , Y , Z هو 54.34999 فذلك يلقى آخر اربع مواقع للكسر العشري فى النتيجة، ويحتوى العنصر ANSWER على 54.3 .

```
.....
ADD X Y Z GIVING ANSWER ROUNDED
```

بمواصفة ROUNDED .. فإننا نحصل على 54.3 فى RESVLت ، آخر رقم يحذف هو 4 ، ولاتقرب النتيجة إلى 54.4.

مثال ٦ - ٤٠ :

ADD HOURS-WORKED TO WEEKLY-TOTAL
MONTHLY-TOTAL ROUNDED
YEARLY-TOTAL ROUNDED

من المسموح به تحديد ROUNDED لبعض الحقول المستقبلية فقط في الحسابات .

٦ - ١٣ تشغيل : السريان الزائد وجزء عند

حدوث خطأ في الحجم

إذا كان جزء الرقم الصحيح integer part للحقل المستقبل أقل من القيمة الخاصة بالنتيجة فسوف يحدث سريان زائد over flow ، أو شرط خطأ في الحجم SIZE ERROR . ويسمح جزء ON SIZE ERROR للمبرمج بأن يحدد مجموعة من العبارات التي لا تنفذ إلا إذا تحقق شرط SIZE ERROR .

مثال ٦ - ٤١ :

ADD WEEKLY-HOURS TO YEARLY-TOTAL-HOURS
ON SIZE ERROR
MOVE EMPLOYEE-ID TO ERROR-LINE-ID
MOVE WEEKLY-HOURS TO ERROR-LINE-HOURS
MOVE "YEARLY HOURS OVERFLOW--HOURS NOT INCLUDED IN TOTAL"
TO ERROR-LINE-MESSAGE
WRITE REPORT-LINE
FROM ERROR-LINE
AFTER ADVANCING 3 LINES

PERFORM COMPUTE-GROSS-PAY
.....

إذا كانت نتيجة جمع WEEKLY- HOURS , YEARLY- TOTAL- HOURS متفقة مع YEARLY- TOTAL- HOURS فتهمل عبارات ON SIZE ERROR بالنسبة لتنفيذ عبارة ADD . لاحظ ان جزء ON SIZE ERROR (مثل AT) AT END... , END- OF- PAGE... ينتهي بنقطة . يتسبب حذف النقطة المرتبة التي تسبق عبارة PERFORM في جعل عبارة PERFORM جزءاً من جزء ON SIZE ERROR ، ونتيجة ذلك هي تنفيذ PERFORM عند تحقق شرط ON SIZE ERROE فقط ؛ بدلا من تنفيذها في كل مرة مطلوبة .

مثال ٦ - ٤٢ :

05 A PIC S99V99 VALUE +50.34.
05 B PIC S99V99 VALUE +50.23.
05 C PIC S99V9 VALUE +12.3.

● ADD A B GIVING C

حيث إن مجموع A , B هو +100.57 .. فإن الجزء الصحيح للنتيجة يكون أكبر من نظيره في الحقل المستقبل C .. ويدون جزء ON SIZE ERROE ، يحذف الكمبيوتر - ببساطة - جزء صحيحاً من النتيجة (من ناحية اليسار) ويستمر في العمل . وعلى هذا يحتوى C على +00.5 ، وهى نتيجة غير صحيحة بالطبع .

- ADD A B GIVING C
ON SIZE ERROR
DISPLAY "SIZE ERROR FOR C-C IS UNCHANGED"

بتحديد جزء ON SIZE ERROE... لا ينقل المجموع المحذوف منه إلى C وتستمر C فى احتوائها على +12.3 . أكثر من ذلك ، تطبع عبارة DISPLAY رسالة خطأ (على ملف نظام خاص SYSOUT بالنسبة لكونل IBM OS/VS) .

خطوط إرشادية خاصة باستخدام عند حدوث خطأ فى الحجم

(١) إذا كان ممكنا ، حدد حجم PIC لعناصر البيانات العددية ، بحيث يكون كبيراً بدرجة كافية ، بحيث تسمح بالنسبة إلى الحسابات التى يجريها البرنامج ، أن يكون السريان الزائد مستحيلاً رياضياً ، ويسمح ذلك بحذف جزء ON SIZE ER- ROE دون خطورة .

(٢) إذا كانت هناك إمكانية بسيطة لحدوث خطأ فى الحجم ، استخدم جزء ON SIZE ERROE .

(٣) عند استخدام ON SIZE ERROE... يجب على الأجزاء التى تلى حدوث الشرط أن :

(أ) تطبع رسالة بها معلومات كافية ، تسمح للشخص المستخدم للكمبيوتر أن يعرف ما حدث .

(ب) ترتب العمل بحيث يترك تشغيل السجلات المنطقية الذى تسبب فى حدوث خطأ الحجم جزئياً أو كلياً .

(ج) تسمح باستمرار التشغيل المعتاد لبقية السجلات المنطقية (لا يتوقف تنفيذ البرنامج كله) .

٦ - ١٤ تشغيل : عبارة الطرح

يعطى شكل ٦ - ٦ صيغتين رئيسيتين لعبارة الطرح SUBTRACT . ونظراً لأن SUBTRACT... FROM... تناظر فى كل الوجوه ADD... TO... فنحتاج الى توضيح معالجة SUBTRACT... FROM... GIVING... فقط .

SUBTRACT { identifier-1 } [literal-1] { identifier-2 } [literal-2] ...
FROM identifier-m [**ROUNDED**] [identifier-n [**ROUNDED**]] ...
[**ON SIZE ERROR** imperative-statement]
(a)

SUBTRACT { identifier-1 } [literal-1] { identifier-2 } [literal-2] ...
FROM { identifier-m } [literal-m]
GIVING identifier-n [**ROUNDED**] [identifier-o [**ROUNDED**]] ...
[**ON SIZE ERROR** imperative-statement]
(b)

شكل (٦ - ٦)

مثال ٦ - ٤٣ :

- SUBTRACT DISCOUNT-AMOUNT FROM CURRENT-PRICE GIVING NEW-PRICE

تطرح المحتويات الحالية للعنصر DISCOUNT- AMOUNT من المحتويات الحالية للعنصر CURRENT- PRICE ، يوضع باقى الطرح فى NEW- PRICE ، يمكن أن يكون الحقل NEW- PRICE حقلا منقحا؛ حيث إنه لايدخل بنفسه فى الحسابات . ولايتأثر أى من DISCOUNT- AMOUNT او CURRENT- PRICE بعملية الطرح .

- SUBTRACT 10 FROM A-GRADE-CUTOFF GIVING B-GRADE-CUTOFF

عادة ماتستخدم الثوابت العددية فى العبارات الحسابية .

- SUBTRACT STATE-TAX
FEDERAL-TAX
FICA-AMOUNT FROM GROSS-PAY
GIVING NET-PAY ROUNDED

NET- PAY هو عنصر البيانات الوحيد الذى تتغير محتوياته . يطرح مجموع FEDER- , FICA- AMOUNT , STATE- TAX , AL- TAX من GROSS- PAY ، ويوضع باقى الطرح فى NET- PAY بعد تقريبه .

- SUBTRACT A B C
FROM D
GIVING E F ROUNDED G
ON SIZE ERROR
PERFORM OVERFLOW-ROUTINE.

يطرح مجموع A , B , C من D ، وتوضع نسخة من النتيجة فى E ، وتقرب نسخة من النتيجة وتوضع فى F ، وتوضع نسخة فى G . إذا تحقق شرط حدوث خطأ فى الحجم أثناء أى من هذه العمليات..فإن المقطع المسمى OVERFLOW- ROU- TINE ينفذ مرة واحدة ، قبل أن يستمر الكمبيوتر فى تنفيذ العبارة التالية .

- 01 SAMPLE-DATA.
05 A PIC S9(3)V9.
05 B PIC S9V99 COMP-3.
05 C PIC ZZZ.99.

SUBTRACT B FROM C GIVING A

الطرح غير صحيح ، لأنه يحاول شمول عنصر منقح (C) فى الحسابات .

- SUBTRACT A B FROM C D GIVING E F

خطأ تكويني : يمكن لعنصر بيانات واحد فقط أن يتبع كلمة FROM .

٦ - ١٥ تشغيل : عبارة الضرب

تستخدم عبارة الضرب MULTIPLY في ضرب عنصرين عدديين ، والعبارة صيغتان .

اضرب فلي

تكوين عبارة أضرب في MULTIPLY... BY... مبين في شكل (٦ - ٧) .

MULTIPLY { identifier-1 } BY identifier-2 [ROUNDED] [identifier-3 [ROUNDED]] ...
[ON SIZE ERROR imperative-statement]

شكل (٦ - ٧)

مثال ٦ - ٤٤ :

- MULTIPLY QUANTITY-PURCHASED BY UNIT-COST

تضرب المحتويات الحالية للعنصر QUANTITY- PURCHASED في المحتويات الحالية للعنصر UNIT- COST ويخزن فيه حاصل الضرب، ولا يتغير QUANTITY- PURCHASED .

- MULTIPLY QUANTITY-PURCHASED
BY UNIT-COST ROUNDED
ON SIZE ERROR
PERFORM EXTENDED-COST-TOO-LARGE

يُقرب حاصل الضرب قبل وضعه في UNIT- COST ، إذا حدث خطأ في الحجم .. ينفذ المقطع -EXTENDED COST- TOO- LARGE مرة واحدة ، قبل أن يستمر الكمبيوتر في تنفيذ العبارة التي تلي عبارة MULTIPLY .

- MULTIPLY .90 BY WHOLESALE-PRICE
RETAIL-PRICE ROUNDED

تضرب محتويات WHOLESALE- PRICE في 90.. وتوضع النتيجة في WHOLESALE- PRICE (مع حذف مواقع الكسر العشري الزائدة) ، ثم تضرب محتويات RETAIL- PRICE في 90، ويوضع حاصل الضرب بعد تقريبه - في RE- TAIL- PRICE .

- MULTIPLY CURRENT-PRICE BY .90

هذا خطأ شائع جداً . ولسوء الحظ.. فإن هذا يوجه الكمبيوتر لضرب محتويات CURRENT- PRICE في 90.. يعتبر وضع النتيجة في الثابت 90 مستحيلاً . وتصحيح ذلك هو مايلي :

MULTIPLY .90 BY CURRENT-PRICE

بافتراض أن المبرمج لا يهتم شيئاً في فقدان المحتويات الاصلية للعنصر CURRENT- PRICE .

اضرب .. في .. معطياً ..

تكوين اضرب .. في .. معطياً .. BY.. GIVING MULTIPLY.. مبين في شكل (٦ - ٨) .

MULTIPLY { identifier-1 } BY { identifier-2 }
{ literal-1 } { literal-2 }
GIVING identifier-3 [ROUNDED] [identifier-4 [ROUNDED]] ...
[ON SIZE ERROR imperative-statement]

شكل (٦ - ٨)

مثال ٦ - ٤٥ :

- MULTIPLY UNIT-PRICE BY QUANTITY-PURCHASED
GIVING EXTENDED-PRICE

تضرب المحتويات الحالية لكل من : QUANTITY- PURCHASED , CURRENT- PRICE ويوضع حاصل الضرب في EXTENDED- PRICE ، ولاتتغير بقية عناصر البيانات الأخرى . يجب أن تكون كل عناصر البيانات عديدة ، ويمكن أن يكون EXTENDED- PRICE منقحا؛ حيث إنه يستقبل نتيجة الحسابات فقط ولا يدخل فيها .

- MULTIPLY A BY B
GIVING C
D ROUNDED
E
ON SIZE ERROR
PERFORM FIX-UP-ROUTINE

تضرب محتويات A في محتويات B ، ويوضع حاصل الضرب في C (مع حذف مواقع الكسبر العشري الأكثر) ، وفي D (مع التقريب) وفي E (مع الحذف) . إذا تحقق شرط حدوث خطأ في الحجم أثناء أى من هذه العمليات ، فينفذ FIX- UP- RONTINE قبل إنتقال الكمبيوتر إلى العبارة التالية .

- MULTIPLY HOURS-WORKED PAY-RATE BY OVERTIME-FACTOR
GIVING OVERTIME-PAY

خطأ تكويني : لا يضرب PAY- RATE , HOURS- WORKED في OVERTIME- FACTOR معا ولا يحدث ذلك بالعبارة التالية كذلك .

MULTIPLY HOURS-WORKED PAY-RATE BY OVERTIME-FACTOR

على عكس ADD... , SUBTRACT... .. يمكن لأمر MULTIY... أن يضرب قيمتين اثنتين فقط . أحد حلول هذه المشكلة يمكن أن يكون على النحو التالي :

MULTIPLY HOURS-WORKED BY PAY-RATE GIVING TEMP
MULTIPLY TEMP BY OVERTIME-FACTOR
GIVING OVERTIME-PAY

حيث TEMP هي منطقة مخزن عمل يعرفها المبرمج ، والحصول على حل أفضل.. انظر القسم السابع عشر من هذا الفصل.

٦ - ١٦ تشغيل : عبارة القسمة

تستخدم عبارة DIVIDE في قسمة محتويات أحد عناصر البيانات العديدة بمحتويات عنصر بيانات عددي آخر . ولها ثلاثة أشكال: أحدها يعطي نتيجتين: خارج القسمة quotient والباقي remainder . ولاتسمح أى صيغة من صيغ عبارة القسمة بالقسمة على صفر . تتسبب محاولة القسمة على صفر في إنهاء البرنامج نهاية غير طبيعية، إلا إذا ما تحدد جزء ON SIZE ERROR ، حيث يحدث في مثل هذه الحالة خطأ في الحجم .

اقسم ... ب ...

يوضح تكوين اقسام ... ب ... INTO... DIVIDE في شكل (٦ - ٩).

```
DIVIDE {identifier-1}
        {literal-1}
      INTO identifier-2 [ROUNDED] [identifier-3 [ROUNDED]] ...
      [ON SIZE ERROR imperative-statement]
```

شكل (٦ - ٩)

مثال ٦ - ٤٦ :

- DIVIDE 100 INTO PERCENTAGE-SCRAPPED

يقسم الثابت العددي بالعنصر PERCENTAGE- SCRAPPED ، الذي تستبدل محتوياته بخارج القسمة . ويجب أن يكون العنصر PERCENTAGE- SCRAPPED عدديا وغير متقا .

- DIVIDE A INTO B
C ROUNDED
D
ON SIZE ERROR
PERFORM ERROR-MESSAGE-ROUTINE

يقسم A بالعنصر B ، وتوضع النتيجة (بعد الحذف من ناحية اليمين) في B ، ويقسم A بالعنصر C ، وتوضع النتيجة (بعد التقريب) في C ، ويقسم A بالعنصر D ، وتوضع النتيجة (بعد الحذف) في D . إذا حدث خطأ في الحجم في أى عملية من العمليات السابقة.. فسوف تنفذ عبارات PERFORM وإلا فإنها تهمل .

- DIVIDE A B C INTO D

خطأ تكويني : يسمح بمعرف واحد فقط على أى جانب من جانبي INTO . (ليس لعبارة ... MULTIPLY وعبارة DI- VIDE... نفس تكوين عبارة -ADD وعبارة ... SUBTRACT) .

اقسم ... ب ... أو على ... معطيا

تكوين عبارة اقسام ب ... أو على معطيا DIVIDE... INTO/BY... GIVING موضع في شكل (٦ - ١٠)

```
DIVIDE {identifier-1} {INTO} {identifier-2}
        {literal-1} {BY} {literal-2}
      GIVING identifier-3 [ROUNDED] [identifier-4 [ROUNDED]] ...
      [ON SIZE ERROR imperative-statement]
```

شكل (٦ - ١٠)

مثال ٦ - ٤٧ :

- **DIVIDE 100 INTO PERCENTAGE-SCRAPPED GIVING FRACTION-SCRAPPED**
تقسم قيمة الثابت 100 بالمحتوى الحالى للعنصر PERCENTAGE- SCRAPPED ، وتخزن النتيجة فى العنصر FRACTION- SCRAPPED . يجب أن تكون كل العناصر عددية ، يمكن أن يكون العنصر FRACTION- SCRAPPED منقحا ، حيث أنه لا يستخدم فى الحسابات لكنه يستقبل النتيجة فقط .

- **DIVIDE PERCENTAGE-SCRAPPED BY 100 GIVING FRACTION-SCRAPPED**

متوافقة تماما فى تأثيرها مع العبارة السابقة .

- **DIVIDE NUMBER-EMPLOYEES INTO TOTAL-SALARIES
GIVING AVERAGE-SALARY
ON SIZE ERROR
PERFORM PRINT-ERROR-MESSAGE**

تقسم المحتويات الحالية للعنصر NUMBER- EMPLOYEES بالمحتويات الحالية للعنصر TOTAL- SALARIES وتوضع النتيجة فى AVERAGE- SALARY (الذى يمكن ان يكون عنصرا منقحا) . لانتغير فيه TOTAL- SALARIES و NUMBER- EMPLOYEE . إذا حدث خطأ فى الحجم .. ينفذ مقطع PRINT- ERROR- MESSAGE .

- **DIVIDE B BY A GIVING C
D ROUNDED
E
ON SIZE ERROR
PERFORM ERROR-ROUTINE**

تقسم المحتويات الحالية للعنصر B على المحتويات الحالية للعنصر A ، وتوضع النتيجة فى C (مع حذف جزء من الكسر إذا كانت هناك حاجة لذلك) ، وفى D (مع التقريب) ، وفى E (مع الحذف) . إذا حدث خطأ فى الحجم أثناء أى من هذه العمليات .. فإن المقطع المسمى ERROR- ROUTINE ينفذ . لا تتغير قيم A ، B ، ويمكن أن تكون العناصر C ، D ، E منقحة إذا كانت هناك رغبة فى ذلك .

- **DIVIDE A INTO B ROUNDED GIVING C**

غير صحيح : يمكن استخدام ROUNDED مع حقول تستقبل خارج القسمة فقط .

اقسم بـ .. أو على معطيا... والباقي

صيغة عبارة القسمة هذه DIVIDE... INTO/BY... GIVING... REMAINDER... موضحة فى شكل (٦ - ١١) .

$$\text{DIVIDE } \left\{ \begin{array}{c} \text{identifier-1} \\ \text{literal-1} \end{array} \right\} \left\{ \begin{array}{c} \text{INTO} \\ \text{BY} \end{array} \right\} \left\{ \begin{array}{c} \text{identifier-2} \\ \text{literal-2} \end{array} \right\}$$

$$\text{GIVING identifier-3 [ROUNDED]}$$

$$\text{REMAINDER identifier-4}$$

$$[\text{ON SIZE ERROR imperative-statement}]$$

شكل (٦ - ١١)

مثال ٦ - ٤٨ :

- $\text{DIVIDE A INTO B GIVING C REMAINDER D}$

يجب أن تكون كل عناصر البيانات عددية ، كما يمكن أن يكون C ، D متقحين . تقسم المحتويات الحالية للعنصر A بمحتويات B ، ويخزن خارج القسمة تخزن في C ؛ أما الباقي فيخزن في D . ويعنى خارج القسمة نتيجة القسمة بأى عدد خانات محدد للكسر العشري في PIC ، الخاصة بالعنصر C ، أما الباقي.. فيعرف على النحو التالى :

الباقي = المقسوم - (خارج القسمة × القاسم (المقسوم عليه)) .

بالنسبة للمثال ، إذا كان A محتويا على 3.3 ، و B محتويا على 10.2 ويمكن للعنصر C أن يحتوى على خانتين للكسر العشري.. فإن C خارج القسمة هي 3.09 والباقي هو 10.2 - (3.3)(3.09) = 0.03.

إذا حدد جزء ROUNDED مع عنصر بيانات GIVING (الذى يستقبل خارج القسمة) ، فإن الباقي يحسب قبل حدوث التقريب . وإذا استخدم جزء ON SIZE ERROR ، وحدث خطأ فى الحجم فى خارج القسمة (عنصر GIVING) .. فلن يتغير أى من عنصر بيانات REMAINDER, GIVING ، وينفذ الجزء الخاص بحدوث الخطأ . أما إذا حدث خطأ فى الحجم بالنسبة لعنصر REMAINDER فقط ، فإن عنصر GIVING يستبدل بخارج القسمة مع عدم تغير عنصر REMAINDER . يستطيع البرنامج أن يقرر ما إذا كان خارج القسمة أو الباقي هو الذى تسبب فى خطأ الحجم، وذلك بالتأكد مما إذا كان حقل خارج القسمة قد تغير (حدث خطأ الحجم قد من خارج القسمة) . تذكر أنه إذا لم يحدد جزء ON SIZE ERROR.. فإن الكمبيوتر سيستمر فى تنفيذ العبارة التالية ، حتى عندما تخزن نتائج خاطئة فى عنصر GIVING أو عنصر REMAINDER.

- $\text{DIVIDE B BY A GIVING C REMAINDER D}$

هذه العبارة تكافئ تماماً العبارة السابقة .

- $\text{DIVIDE 24 INTO TOTAL-HOURS GIVING NUMBER-OF-DAYS}$
 $\text{REMAINDER HOURS-LEFT-OVER}$

خارج القسمة هو عدد الأيام الكاملة المثلثة بواسطة TOTAL- HOURS ، ويمثل REMAINDER أى جزء متبقياً من الأيام بالساعات .

- DIVIDE BALANCE-DUE BY AMOUNT-EACH-PAYMENT
GIVING NUMBER-OF-FULL-PAYMENTS
REMAINDER AMOUNT-OF-LAST-PAYMENT

فى تطبيقات التسليف التجارية عادة ماتختلف آخر دفعة مدفوعة عن بقية الدفعات . تحسب عبارة DIVIDE عدد الدفعات المعتادة بقيمة الدفعة الأخيرة (الأقل) .

- DIVIDE A INTO B GIVING C ROUNDED REMAINDER D ROUNDED

خطأ تكويني : يمكن استخدام ROUNDED مع عنصر GIVING فقط ، وليس مع عنصر REMAINDER.

- DIVIDE PAYMENT-AMOUNT INTO BALANCE-DUE
REMAINDER LAST-PAYMENT

هذا خطأ شائع ، يجب استخدام جزء GIVING عندما يتحدد جزء REMAINDER . إذا لم يكن خارج القسمة مهما فعلا ، فيمكن استخدام الوسيلة التالية :

DIVIDE PAYMENT-AMOUNT INTO BALANCE-DUE
GIVING DUMMY-FIELD
REMAINDER LAST-PAYMENT

حيث DUMMY- FIELD معروف فى مخزن العمل (لكنه لا يستخدم على الإطلاق) .

٦ - ١٧ تشغيل : عبارة الحساب

تسمح عبارة الحساب COMPUTE بتحديد عديد من العمليات الحسابية مرة واحدة . وعادة ماتنتج عنها برامج هدف أكثر كفاءة عن إجراء الحسابات باستخدام عبارات ADD , SUBTRACT , MULTIPLY , DIVIDE . وعلى هذا .. يجب أن نكتب أى حسابات تشمل أكثر من عملية حسابية واحدة باستخدام COMPUTE .

ولعبارة الحساب شكل واحد فقط (شكل ٦ - ١٢) .

COMPUTE identifier-1 [ROUNDED]
[identifier-2 [ROUNDED]] ...
= arithmetic-expression
[ON SIZE ERROR imperative-statement(s)]

شكل (٦ - ١٢)

يشبه التعبير الحسابي arithmetic-expression في عبارة COMPUTE التعبيرات التي تحدث في الجبر الأولى. وتتكون من عناصر بيانات عددية، أو ثوابت عددية مخلوطة بمؤثر واحد أو أكثر من المؤثرات الحسابية arithmetic operators في الكويل .

+ الجمع / القسمة
- الطرح ** الأس (مرفوعاً للقوة)
* الضرب

هذه هي مؤثرات ثنائية binary operators لأنها تتطلب عاملين اثنين two operands تعملان عليهما. يسمح الكويل كذلك باستخدام إشارة سالبة أحادية unary minus sign لنفي القيمة العددية . لا يمكن أن تكون العناصر الموجودة في الناحية اليمنى من علامة التساوي ، التي تدخل الحسابات منقحة ، أما عناصر البيانات الموجودة في الناحية اليسرى من علامة التساوي، التي تستقبل النتيجة.. فمن الممكن أن تكون منقحة . في عبارات COMPUTE .. يجب أن يسبق المؤثر الحسابي ويتبعه فراغ، أو يسبقه قوس مغلّق، أو يتبعه قوس مفتوح .

مثال ٦ - ٤٩ :

COMPUTE A = B + C

تكافئ

ADD B C GIVING A

COMPUTE A = B - C

SUBTRACT C FROM B GIVING A

تكافئ

COMPUTE A = B * C

تكافئ

MULTIPLY B BY C GIVING A

COMPUTE A = B / C

تكافئ

DIVIDE B BY C GIVING A

COMPUTE A = B ** C

ليس لها مكافئ في عبارة أخرى. $B^{**}C$ تمثل B مرفوعاً للقوة C أي B^C . فمثلاً $B^{**}3$ هي تكعيب.

COMPUTE A = - B

(إشارة السالب الأحادية) تكافئ

SUBTRACT B FROM ZERO GIVING A

يمكن استخدام الأقواس للتحكم فى ترتيب العمليات داخل عبارة COMPUTE : تنفذ العمليات الموجودة فى الأقواس الداخلية جدا أولا ، تليها العمليات الموجودة بين القوسين التاليين ، وهكذا . وفى حالة عدم وجود الأقواس ، يتبع المترجم القواعد التالية :

(١) كل إشارات السالب الفردية تؤدي أولا من اليسار .. لليمين .

(٢) كل عمليات * و / تؤدي بعد ذلك من اليسار لليمين .

(٣) كل عمليات + و - تؤدي بعد ذلك من اليسار لليمين .

(٤) كل عمليات + و - تؤدي فى النهاية من اليسار لليمين .

مثال ٦ - ٥٠ :

- COMPUTE A ROUNDED = (B + C) * (D + E)

بالترتيب : (١) يحسب جمع B , C (٢) يحسب جمع E , D (٣) نتيجة (١) و (٢) تضرب فى بعضهما (٤) يقرب حاصل الضرب (٥) يوضع حاصل الضرب المقرب فى A . ويجب أن تكون العناصر B , C , D , E عددية غير منقحة ، ويمكن أن يكون A منقحا .

- COMPUTE A ROUNDED
B
C
= A + B * C / D
ON SIZE ERROR
PERFORM PRINT-ERROR-MESSAGE

فى غياب الأقواس ، يحدث يمايلى : (١) يحسب B*C (٢) تقسم نتيجة (١) على D (٣) تجمع نتيجة (٢) على A (٤) توضع نتيجة (٣) فى A (مع التقريب) ، (٥) توضع نتيجة (٣) فى B (مع الحذف) ، (٦) توضع نتيجة (٣) فى C (مع الحذف) . إذا حدث خطأ فى الحجم أثناء تخزين النتيجة فى A أو B أو C فإن المقطع PRINT- ERROR- MESSAGE ينفذ .

- COMPUTE GROSS-PAY = (40 * RATE) +
(1.5 * RATE * (HOURS-WORKED - 40))

هذه هى صيغة تقليدية فى حالة العمل وقت إضافى . يدفع المعدل المعتاد على أول 40 ساعة عمل (40*RATE) . ويدفع مرة ونصف من المعدل المعتاد (1.5*RATE) لساعات العمل الأكثر من 40 ساعة (HOURS - WORKED - 40) . وإجمالى الأجر GROSS- PAY هو مجموع القيمتين المدفوعتين .

- COMPUTE NUMBER-EMPLOYEES = NUMBER-EMPLOYEES + 1

هذه العبارة التى تبدو غريبة مكافئة لمايلى :

ADD 1 TO NUMBER-EMPLOYEES

- COMPUTE QUANTITY-ON-HAND =

QUANTITY-ON-HAND	(100)
+ QUANTITY-RECEIVED	(10)
+ QUANTITY-RETURNED-TO-US	(20)
- QUANTITY-SHIPED	(30)
- QUANTITY-SCRAPPED	(40)

تضاف محتويات QUANTITY- RECEIVED , QUANTITY- RETURNED- TO- US إلى القيمة الحالية للعنصر QUANTITY- ON- HAND ، ثم تطرح قيمة QUANTITY-SHIPED , QUANTITY- SCRAPPED من النتيجة . وتوضع النتيجة النهائية QUANTITY- ON- HAND مدمرة بذلك القيمة الأصلية التي كانت بها . إذا كان لكل عنصر بيانات القيمة الموجودة بين قوسين أمامه عند البداية ، تصبح المحتويات النهائية للعنصر QUANTITY- ON- HAND هي 60. وتظل بقية عناصر البيانات الأخرى كما هي دون تغيير .

- COMPUTE AVERAGE-PAY = $(\text{SALARIED-TOTAL} + \text{NONSALARIED-TOTAL}) / \text{NUMBER-EMPLOYEES}$

يعد استخدام الأقواس ضرورياً هنا ؛ فبدونها .. يقسم NONSALARIED- TOTAL على NUMBER- EMPLOY- EES، وتضاف النتيجة إلى SALARIED- TOTAL ، فتنتج عن هذا قيمة خاطئة لتوسط مايدفع .

- COMPUTE CLEARANCE-PRICE ROUNDED = $\text{REGULAR-PRICE} - .15 * \text{REGULAR-PRICE}$

ليس هناك حاجة للأقواس هنا ، حيث أن الضرب يسبق الطرح على أية حال . وفيمايلي صيغة أبسط لنفس الحسابات .

COMPUTE CLEARANCE-PRICE ROUNDED = $.85 * \text{REGULAR-PRICE}$

- COMPUTE C = $(A ** 2 + B ** 2) ** .5$

الرفع إلى القوة 5. هو نفسه مثل أخذ الجذر التربيعي ، وعلى هذا فيحسب الجذر التربيعي لمربع الكمية A مضافا إليه مربع الكمية B . والتمثيل الجبري لهذا هو مايلي :

$$c = \sqrt{a^2 + b^2}$$

مثال ٦ - ٥١ :

افرض أن الدفعة الشهرية لقرض تحسب طبقاً لأساس المبلغ، والمبالغ المسددة من الفائدة، والموازنة الجديدة .

COMPUTE INTEREST-PAID = $\text{OLD-BALANCE} * (\text{ANNUAL-INTEREST-RATE} / 12)$

COMPUTE PRINCIPAL-PAID = $\text{PAYMENT-AMOUNT} - \text{INTEREST-PAID}$

COMPUTE NEW-BALANCE = $\text{OLD-BALANCE} - \text{PRINCIPAL-PAID}$

لاحظ أن معدل الفائدة السنوية يقسم على 12 للحصول على معدل الفائدة الشهرية .

مثال ٦ - ٥٢ :

(استخدام غير صحيح لعبارة COMPUTE) (أ) ثمة خطأ شائع للمبتدئ وهو فشله في ترك فراغ واحد على الأقل، قبل وبعد المؤثر الحسابي ، ولا يكون محاطاً بـ ، أو «أو قبل وبعد علامة التساوي» .

COMPUTE A = B+C
COMPUTE A = B + C*D
COMPUTE A = (A+B)/(A-B)

كلها غير صحيحة ، وذلك بالرغم من أن

“COMPUTE A = (A + B)/(A - B)”

صحيحة

(ب) فقدان فراغات محيطة يجعل إشارة السالب تبدو مثل الشرطة .

COMPUTE A = B-C

يبحث المترجم عن عنصر بيانات اسمه B-C ، ولا يفسر هذا كعملية طرح .

(جـ) هناك خطأ يتكرر ، وهو الفشل في الحصول على الحسابات المرجوة بسبب عدم وجود الأقواس . ولا يوجد على الإطلاق أى عيب في استخدام الأقواس - فإذا كان هناك أى شك فيما إذا كان هناك حاجة للأقواس أم لا ، فاستخدمها وأنت متأكد .

COMPUTE NEW-BALANCE = OLD-BALANCE
- PAYMENT-AMOUNT
+ OLD-BALANCE *
ANNUAL-INTEREST-RATE / 12

تكون صحيحة ، إلا أن مايلي صحيحاً كذلك ، له نفس الكفاءة .

COMPUTE NEW-BALANCE =
OLD-BALANCE
- (PAYMENT-AMOUNT
- (OLD-BALANCE * (ANNUAL-INTEREST-RATE / 12)))

لاحظ كيف تقع الأقواس في أزواج متوازنة ، أى لا بد من غلق كل قوس . ويمثل عدم التوازن (ظهور قوس مفتوح بدون آخر مغلق أو قوس مغلق بدون آخر مفتوح) خطأ شائع الحدوث .

٦ - ١٨ الكفاءة في الحسابات الرياضية

بصفة عامة.. يمكن الحصول على برنامج هدف أكثر كفاءة عندما :

(١) تكون لعناصر البيانات المشمولة في الحسابات PICTURE محددة فيها العلامة العشرية .

(٢) يكون لعناصر البيانات المشمولة في الحسابات نفس الاستخدام USAGE .

- (٣) لعناصر البيانات المستخدمة في الحسابات استخدام من نوع COMP أو COMP-3 .
- (٤) تكون لحقول COMP-3 (في كويل IBM OS/VS) عدد أرقام فردى (وبصفة خاصة عند استخدام ON SIZE ER- (ROR) .
- (٥) يكون أحد الحقول في سجل منطقي، استخدام DISPLAY ، ويكون مشمولا في عديد من الحسابات ؛ فينقل الحقل إلى منطقة مخزن عمل، تكون لها استخدام COMP أو COMP-3 وتستخدم نسخة مخزن العمل هذه في الحسابات . (أى يحدث تحويل واحد للاستخدام بدلا من عديد من التحويلات) .
- (٦) تستخدم S في PICTURE العددية المشمولة في الحسابات (إلا إذا ما كان المطلوب هو القيمة المطلقة) .
- (٧) يتم تجنب مواصفة ON SIZE ERROR بالاختيار المناسب للصون PICTURES .

٦ - ١٩ زهديات كتابة شفرة اضافية

لجزء الاجراءات

(ارجع إلى القسم ٦ - ١)

- (١١) استخدم STOP RUN مرة واحدة فقط في البرنامج .
- (١٢) العبارات الطويلة تجزأ عند بدء الجزء ثم ترحل في السطر التالي .
- (١٣) ضع الأجزاء مثل AT END- OF- PAGE , AT END- OF- PAGE , ON SIZE ERROR مرحلة على السطور الخاصة بها .
- (١٤) أهبط العبارات أو أجزاء العبارات المتشابهة رأسيا مع بعضها البعض .
- (١٥) استخدم ثابتاً إذا كنت متأكداً أنه لن يتغير على الإطلاق أثناء فترة استخدام البرنامج .

مثال ٦ - ٥٣ :

- MOVE ZERO TO NUMBER-OF-EMPLOYEES

كما توضع NUMBER- OF- EMPLOYEES مساوية صفرا في بداية تنفيذ البرنامج دائما .. فإن استخدام الثابت الاستعاري ZERO يكون مناسباً .

- MULTIPLY REGULAR-PRICE BY .08 GIVING EMPLOYEE-DISCOUNT

يمكن أن يتغير معدل خصم العميل من 08. إلى قيمة أخرى ، ويجب ألا يستخدم ثابت هنا . بدلا من ذلك... يجب أن يعرف المبرمج عنصر بيانات في مخزن العمل على النحو التالي :

WORKING-STORAGE SECTION.

01 PROGRAM-CONSTANTS.

05 EMPLOYEE-DISCOUNT-RATE PIC S999
VALUE = .08.

.....

ويجب أن يستبدل MULTIPLY السابقة بما يلي :

MULTIPLY REGULAR-PRICE BY EMPLOYEE-DISCOUNT-RATE GIVING
EMPLOYEE-DISCOUNT

والآن ، إذا وجب تغيير معدل خصم العميل .. فلن يكون هناك حاجة الا إلى تغيير VALUE واحدة فقط ؛ بدلا من تغيير الثابت نفسه في كل موقع يظهر فيه جزء الإجراءات .

اسئلة مراجعة

- ٦ - ١ ما الغرض من جزء الإجراءات ؟
- ٦ - ٢ أشرح هيكل جزء الإجراءات بتعريف : القسم ، والمقطع ، والجملة ، والعبارة ، والفعل .
- ٦ - ٣ ناقش نمطيات كتابة الشفرة لكتابة جزء الإجراءات .
- ٦ - ٤ ما وظيفة عبارة الفتح OPEN ؟
- ٦ - ٥ وضح كل حالة من حالات عبارة الفتح : EXTEND , I/O , OUTPUT , INPUT .
- ٦ - ٦ ناقش استخدام عبارة الفتح في فتح كل الملفات في فتح كل ملف على حدة .
- ٦ - ٧ ماذا يحدث إذا ما أشار البرنامج إلى منطقة سجل منطقي الملف قبل فتح الملف ، أو بعد إغلاق الملف ؟
- ٦ - ٨ ما وظيفة عبارة الإغلاق CLOSE ؟
- ٦ - ٩ قارن بين إغلاق كل الملفات بعبارة إغلاق واحدة ، وبين إغلاق كل ملف بعبارة إغلاق مستقلة به .
- ٦ - ١٠ ناقش كيف يمكن فتح وإغلاق ملف عديد من المرات بواسطة نفس البرنامج .
- ٦ - ١١ ما وظيفة WITH LOCK في عبارة CLOSE ؟
- ٦ - ١٢ ما الشريط متعدد الملفات ؟
- ٦ - ١٣ وضح استخدام STANDARD LABELS مع الشريط متعدد الملفات .
- ٦ - ١٤ ما الملف متعدد الحجم ؟
- ٦ - ١٥ ماذا يعنى عمل مفتاح للحجم volume switching ؟
- ٦ - ١٦ ما الغرض من عبارة القراءة READ ؟
- ٦ - ١٧ وضح الغرض من جزء AT END في عبارة القراءة .
- ٦ - ١٨ ما شرط نهاية الملف ؟ ومتى يتحقق هذا الشرط ؟
- ٦ - ١٩ لماذا يكون من الضروري عدم تشغيل سجل منطقي بعد حدوث نهاية الملف ؟
- ٦ - ٢٠ لماذا يختار بعض المبرمجين عمل تشغيل كل السجلات المنطقية ، باستخدام نسخ السجلات المنطقية في مخزن العمل ؟
- ٦ - ٢١ وضح استخدام READ... INTO .
- ٦ - ٢٢ ماذا يعنى شعاره اقرأ ملفاً واكتب سجلاً ؟
- ٦ - ٢٣ ما الغرض من عبارة الكتابة WRITE ؟
- ٦ - ٢٤ ماذا يوجد في منطقة سجل منطقي الملف فور إمام عملية كتابة WRITE بنجاح ؟

- ٦ - ٢٥ ما بعض مميزات بناء سجلات منطقية في منطقة مخزن عمل، ثم استخدام WRITE... FROM - بعد ذلك - في نقلها إلى ذاكرة احتياطية للمخرجات ؟
- ٦ - ٢٦ إذا لم يمكن استخدام WRITE... FROM , READ... INTO مع سجلات متغيرة الأطوال .. كيف يستطيع البرنامج تشغيل مثل هذه السجلات في مناطق مخزن العمل ؟
- ٦ - ٢٧ ناقش الطرق المختلفة للتحكم في فراغات الطابع للملفات التي يتم إخراجها عن طريق الطابع .
- ٦ - ٢٨ ماذا يحدث إذا لم تشمل الكتابة في ملف على الطابع جزءاً لمسافات الطابع ؟
- ٦ - ٢٩ اذكر بعض الأمثلة لمواقف، يجب استخدام عنصر بيانات فيها لتحديد عدد الأسطر في جزء BEFORE/AFTER ADVANCING من عبارة الكتابة .
- ٦ - ٣٠ ما المعاني الممكنة لـ BEFORE/AFTER ADVANCING PAGE ؟
- ٦ - ٣١ وضح كيف يمكن استخدام قنوت تحكم العربية المحددة، في مقطع الأسماء الخاصة، في التحكم في فراغات الطابع WRITE .
- ٦ - ٣٢ وضح مناطق صفحة منطقية التي تعرف بجزء LINAGE .
- ٦ - ٣٣ اذكر الشرطين الذي يتفقد تحتها مقطع AT END- OF- PAGE كجزء من عبارة الكتابة .
- ٦ - ٣٤ ماذا يمكن أن يحدث إذا لم تتسب الأجزاء التالية بطريقة مناسبة: LINAGE ، وجزء BEFORE/ AFTER ، وجزء AT END- OF- PAGE ؟
- ٦ - ٣٥ ما السريان الزائد التلقائي للصفحة ؟
- ٦ - ٣٦ عرف العبارة الشرطية والعبارة الأمرية .
- ٦ - ٣٧ متى تكون عبارة القراءة READ شرطية ؟ ومتى تكون أمرية ؟
- ٦ - ٣٨ متى تكون عبارة الكتابة WRITE شرطية ؟ ومتى تكون أمرية ؟
- ٦ - ٣٩ مانوع معلومات نظام التشغيل الذي يمكن الحصول عليه بعبارة ACCEPT ؟ وكيف يمكن استخدام هذه المعلومات ؟
- ٦ - ٤٠ اذكر بعض الأمثلة التي يمكن استخدام ACCEPT فيها لإدخال معلومات من النهاية الطرفية المتاحة لمشغل الكمبيوتر.
- ٦ - ٤١ متى يجب ألا تستخدم ACCEPT بدلاً من إعداد ملف معتاد باستخدام FD , OPEN , REEAD , CLOSE ؟ SELECT
- ٦ - ٤٢ ما المقصود ببطاقات التحكم ؟
- ٦ - ٤٣ كيف يمكن استخدام عبارة DISPLAY في تصحيح برنامج ؟
- ٦ - ٤٤ متى يمكن استخدام DISPLAY لرسائل الخطأ (بدلاً من ملف المخرجات المعتاد المطبوع باستخدام CLOSE, (WRITE, OPEN, FD SELECT ؟
- ٦ - ٤٥ اذكر بعض الأمثلة التي يمكن استخدام DISPLAY فيها؛ للاتصال مع مشغل الكمبيوتر عبر نهايته الطرفية .
- ٦ - ٤٦ بالإضافة إلى نقل بيانات.. ما الوظائف الأخرى التي تؤديها عبارة MOVE ؟
- ٦ - ٤٧ عرف : (أ) الحقل الراسل . (ب) الحقل المستقبل (ج) التحويل (د) المله بفراغات أو أصفار (هـ) الحذف (و) التنقيح .

- ٤٨ - ٦ ما هي قواعد الحذف والملاء المستخدمة (أ) مع البيانات العددية (ب) مع البيانات غير العددية عندما يحدث نقل لها ؟
- ٤٩ - ٦ ماذا يحدث في الكويل عندما تنقل مجموعة عناصر ؟ اذكر بعض الأخطاء شائعة حدوث عند نقل مجموعة عناصر ؟
- ٥٠ - ٦ متى يكون تنفيذ نقل مجموعة عناصر آمناً ؟
- ٥١ - ٦ صف خواص خليط العناصر الفردية الذي لا يكون صحيحاً في عبارة النقل و اذكر عدة أمثلة لنقل فردى غير صحيح.
- ٥٢ - ٦ ما وظيفة عبارة STOP RUN ؟
- ٥٣ - ٦ لماذا يجب أن تفلق كل الملفات قبل إيقاف تنفيذ البرنامج ؟
- ٥٤ - ٦ ماذا يحدث عند تنفيذ العبارة "STOP GIVE ME A PEN" ؟
- ٥٥ - ٦ ما الغرض من عبارة PERFORM ؟
- ٥٦ - ٦ ميز بين الترتيب الواقعي لعبارات جزء الإجراءات وترتيبها المنطقي ؟
- ٥٧ - ٦ ما دورة loop البرنامج ؟
- ٥٨ - ٦ وضع بالتفصيل كيفية عمل عبارة PERFORM بسيطة .
- ٥٩ - ٦ وضع بالتفصيل كيفية عمل عبارة PERFORM... UNTIL .
- ٦٠ - ٦ ماذا يحدث إذا تحقق شرط PERFORM... UNTIL فجأة في منتصف تنفيذ مقطع PERFORM ؟
- ٦١ - ٦ ما برنامج التتبع ؟
- ٦٢ - ٦ ما النسخة الدائمة ؟
- ٦٣ - ٦ ما الغرض من عبارة ADD ؟
- ٦٤ - ٦ ميز بين تأثيرات الحذف التقليدي وجزء ROUNDED عند الحصول على نتائج كسرية أثناء حسابات الكمبيوتر .
- ٦٥ - ٦ ما معنى السريان الزائد ؟
- ٦٦ - ٦ وضع استخدام جزء ON SIZE ERRO ، في العبارة الحسابية .
- ٦٧ - ٦ ماذا يمكن عمله لجعل مقطع ON SIZE ERROR غير ضروري في البرنامج ؟
- ٦٨ - ٦ اذكر خطأ إرشادية لكتابة مقطع ON SIZE ERROR .
- ٦٩ - ٦ أين يمكن استخدام عناصر عددية منقحة في العبارات الحسابية ؟
- ٧٠ - ٦ وضع الغرض من عبارة MULTIPLY . كيف يختلف تكوينها عن تكون عبارة ADD ، وعبارة SUBTRACT ؟
- ٧١ - ٦ ما الغرض من عبارة DIVIDE ؟
- ٧٢ - ٦ عرف : (أ) خارج القسمة (ب) الباقي .
- ٧٣ - ٦ ماذا يحدث في الكويل ، إذا حاولت القسمة على صفر ؟
- ٧٤ - ٦ اذكر بعض ميزات استخدام عبارة COMPUTE ؟
- ٧٥ - ٦ اذكر المؤثرات الخمس الثنائية المستخدمة في عبارة COMPUTE .
- ٧٦ - ٦ ماذا تعني إشارة السالب الأحادية ؟
- ٧٧ - ٦ ما دور الأقواس في عبارة COMPUTE ؟
- ٧٨ - ٦ ما الترتيب التقليدي لتنفيذ العمليات الحسابية إذا لم تستخدم الأقواس ؟
- ٧٩ - ٦ اذكر بعض الأخطاء المعتادة في استخدام COMPUTE ؟

- ٦ - ٨٠ ناقش مايمكن عمله فى الكوبل لجعل العمليات الحسابية ذات كفاءة بقدر الإمكان .
- ٦ - ٨١ متى يمكن أن تستخدم الثوابت بأمان فى البرنامج ؟
- ٦ - ٨٢ ما الطريقة التى يمكن استخدامها للإحلال محل الثوابت فى جزء الإجراءات ؟ ومتى يجب استخدامها ؟ لماذا تجعل هذه الطريقة البرنامج أسهل فى تعريفه فيما بعد ؟

مسائل حلولة

- ٦ - ٨٣ وضع العبارة : «تشمل معظم خوارزميات الأعمال تكرارا لدورة مدخلات - تشغيل - مخرجات - حتى نهاية الملف» الخوارزم algorithm عبارة عن مجموعة ليست غامضة من الخطوات التفصيلية، لعمل نشاط معين أو حل مشكلة معينة. تشمل مشاكل الأعمال التقليدية تشغيلاً لكل السجلات المنطقية الموجودة في ملف المدخلات، وهذه المشاكل مبنية على ذلك حول دورة رئيسية main loop يحدث فيها: (١) يكون السجل التالى مدخلات (٢) يتم تشغيل لسجل المدخلات المنطقى الذى دخل توا (٣) اذا كان ممكنا ، تخرج نتائج تشغيل السجل المنطقى . وتكرر الخطوات الثلاث من (١) الى (٣) حتى تصل لنهاية الملف أثناء محاولة تنفيذ الخطوة رقم (١) .
- ٦ - ٨٤ يقرأ أحد البرامج ملف مدخلات، به سجلات تاريخ المبيعات، ويضيفها إلى ملف آخر موجود . وضع كيف يفتح كل من الملفين.

OPEN	INPUT	SALES-HISTORY-TRANSACTIONS
	EXTEND	SALES-HISTORY-MASTER

- جزء EXTEND ، يسمح بإضافة سجلات جديدة الى نهاية ملف تتابعى موجود فعلاً .
- ٦ - ٨٥ افرض فى المسألة السابقة.. أنه من المطلوب حذف كل المحتويات السابقة للملف واستبدالها بسجلات جديدة كلية . اكتب عبارة الفتح اللازمة لذلك .

OPEN	INPUT	SALES-HISTORY-TRANSACTIONS
	OUTPUT	SALES-HISTORY-MASTER

عندما يفتح ملف تتابعى موجود فعلاً كملف مخرجات ، تحذف كل السجلات المنطقية السابقة منه، وينتج صيغة جديدة تماماً للملف الناتج من كتابة WRITING سجلات منطقية فيه .

- ٦ - ٨٦ حدد الخطأ فى جزء الإجراءات التالى :

```
FD INPUT-FILE ...
01 INPUT-RECORD ...

FD REPORT-FILE ...
01 REPORT-LINE ...
```


PROCEDURE DIVISION.

MOVE SPACES TO INPUT-RECORD
 WRITE REPORT-LINE FROM WS-HEADING-LINE
 AFTER ADVANCING PAGE
 OPEN INPUT INPUT-FILE
 OUTPUT REPORT-FILE

يوضح هنا خطآن شائعان، هما :

(١) تتصلل MOVE SPACES TO INPUT- RECORD بمنطقة سجل منطقي للملف قبل فتح الملف، ويتسبب ذلك في حدوث أخطاء جسيمة دائماً ، حيث لا تتاح ذاكرات الملف الاحتياطية بعد للبرنامج .

(٢) تتصلل WRITE REPORT- LINE FROM منطقة سجل منطقي قبل فتح الملف (حيث إن WRITE... FROM تكافئ عبارة MOVE يتبعها عبارة WRITE) وأسوأ من ذلك .. يتم في محاولة إخراج سجل قبل فتح الملف، حيث يتسبب هذا الخطأ الأخير في إنهاء البرنامج نهاية غير طبيعية .

٦ - ٨٧ تنقل كل من CLOSE , WRITE , READ, OPEN مفتاح حالة مكون من 2 بايت في منطقة FILE STATUS للملف (بافتراض أن FILE STATUS معرفة في الملف).

ما قيمة مفتاح الحالة التي تحدد تنفيذا ناجحاً لأي من هذه العبارات ؟

في كويل IBM SO/VS .. دائماً ما يحدد 00 اتعاماً ناجحاً لعبارة مدخلات أو مخرجات . يمكن استخدام PIC 99 أو PIC XX لمفتاح الحالة .

٦ - ٨٨ حدد الخطأ فيما يلي :

```
05 FILE-STATUS-AREA PIC XX.
.....
OPEN INPUT SAMPLE-FILE
IF FILE-STATUS-AREA EQUAL 00 ...
```

منطقة مفتاح الحالة المعرفة بالصورة PIC XX تكون غير عديدة ، وعلى هذا .. يجب استخدام ثابت غير عددي :

```
IF FILE-STATUS-AREA EQUAL "00" ...
```

٦ - ٨٩ ماذا يمكن أن يتسبب في فشل عبارة CLOSE (ويضع رمزاً غير صفري في منطقة FILE STATUS) ؟

إذا كانت للملف عناوين منطقية .. فيمكن أن يتسبب فشل نظم المكونات في أخطاء مدخلات ومخرجات غير قابلة للتصليح، وذلك أثناء تشغيل العنوان . وهناك إمكانية أخرى وهي أنه قد لا يفتح الملف فتح بنجاح ، وفي هذه الحالة .. لن يفلح بنجاح .

٦ - ٩٠ بين كيف تعلق ٢ ملفات مقلدا: (أ) وقت التنفيذ (ب) عدد البايت من الذاكرة .

(أ)
CLOSE INVENTORY-TRANSACTION-FILE
INVENTORY-MASTER-FILE
INVENTORY-UPDATE-FILE

عندما نعلق عدة ملفات بعبارة CLOSE واحدة .. يقل وقت التنفذ (إلا أنه هناك حاجة إلى مزيد من ذاكرة الكمبيوتر).

(ب)
CLOSE INVENTORY-TRANSACTION-FILE
CLOSE INVENTORY-MASTER-FILE
CLOSE INVENTORY-UPDATE-FILE

يقل إغلاق كل ملف على حدة من استخدام الذاكرة، إلا أنه يستغرق وقتا أطول .
٦ - ٩١ افترض أن هناك برنامجا يقوم بتشغيل ملفين على نفس الشريط (شريط متعدد الملفات) . يفتح البرنامج الملف الأول ويشغله، ثم يعلق ويفتح الملف الثاني . اكتب عبارتي الفتح وعبارة إغلاق واحدة لهذا البرنامج .

OPEN INPUT FIRST-FILE-ON-TAPE
.....
CLOSE FIRST-FILE-ON-TAPE WITH NO REWIND
OPEN INPUT SECOND-FILE-ON-TAPE

٦ - ٩٢ عندما يقوم أحد البرامج بتشغيل ملف متعدد الأحجام تتابعيا .. فعادة مايعامل نظام التشغيل تحويل أحجام تلقائيا automatic volume switching . وضع كيفية عمل ذلك (أ) الملف مدخلات (ب) ملف مخرجات .

(أ) يبدأ البرنامج قراءة سجلات منطقية من أول حجم من أحجام الملف المتعدد . إذا كان هناك 500 سجل منطقي على أول حجم للملف .. فإن تنفيذ عبارة READ رقم 501 يتسبب في تحويل الحجم ، الذي يقوم فيه نظام التشغيل بمايلي: (١) التأكد من أن الحجم الثاني من الملف موضوع بطريقة مناسبة (٢) تشغيل أى سجلات عناوين نمطية للحجم الثاني (٣) جعل أول سجل منطقي من الحجم الثاني للملف متاحا للبرنامج .

(ب) عندما يفتح ملف مكون من عدة أحجام كمخرجات ، فيوضع الملف عند بداية أول حجم . عند ذلك يبدأ البرنامج كتابة سجلات منطقية في الحجم الأول . عندما ينفذ البرنامج عبارة WRITE ، ولايجد مكانا متبقيا للتسجيل على الحجم الأول : فيقوم نظام التشغيل بعمل مايلي تلقائيا: (١) إنهاء تشغيل أى عناوين نمطية للحجم الأول (٢) بدء تشغيل العناوين النمطية للحجم الثاني (٣) كتابة سجل منطقي في بداية الحجم الثاني .

٦ - ٩٣ أربط عبارة CLOSE... UNIT بتغيير الحجم تلقائيا .

لقد رأينا في المسألة السابقة .. كيف يبدأ نظام التشغيل في تغيير الحجم تلقائيا، عندما يصل إلى نهاية الحجم أثناء

التشغيل المعتاد للملف . ويمكن استخدام عبارة CLOSE... UNIT في إجبار تغيير الحجم عند أى نقطة اختيارية في البرنامج . ويمكن أن يكون هذا مفيداً عندما يتم إنتاج ملف متعدد الحجم ؛ لأنه يسمح للبرنامج بالتحكم مباشرة في عدد السجلات المنطقية التي تكتب في كل حجم .

٦ - ٩٤ حدد الخطأ فيما يلي :

```
OPEN INPUT SAMPLE-FILE
READ SAMPLE-FILE
  AT END
    MOVE "T" TO END-OF-FILE-SW
MOVE SAMPLE-RECORD TO OUTPUT-AREA
WRITE OUTPUT-RECORD
  FROM OUTPUT-AREA.
```

هناك أخطاء عديدة : (١) لا توجد نقطة تحدد نهاية جزء AT END . (٢) تنفذ WRITE OUTPUT- RECORD بدون فتح ملف مخرجات . (٣) يفشل البرنامج في اختبار نهاية الملف قبل محاولته MOVE SAMPLE- RECORD TO OUTPUT- AREA . إذا حدثت نهاية الملف ، فلن يوجد SAMPLE- RECORD لتشغيله (ويدخل البرنامج قيمة نفايا في التشغيل).

٦ - ٩٥ الهدف من جزء الإجراءات التالي هو نسخ ملف به سجلات على بطاقات من 80 عموداً في ملف قرص . صحح الأخطاء

PROCEDURE DIVISION.

```
OPEN INPUT CARD-FILE
MOVE "NO" TO END-OF-FILE
PERFORM COPY-A-RECORD
  UNTIL END-OF-FILE EQUAL "YES"
CLOSE DISK-FILE
```

COPY-A-RECORD.

```
READ CARD-FILE
  AT END MOVE "YES" TO END-OF-FILE
MOVE CARD-RECORD TO DISK-RECORD
WRITE DISK-RECORD
```

الأخطاء بترتيب حدوثها ، هي : (١) لم يفتح DISK- FILE . (٢) لم يغلَق CARD- FILE . (٣) لا توجد STOP RUN . (٤) لا توجد نقطة بعد جزء AT END (٥) لا يوجد اختبار لنهاية الملف، قبل محاولة نقل MOVE ، وكتابة WRITE سجل بطاقة . الصيغة المصححة هي كما يلي :

PROCEDURE DIVISION.

```

OPEN INPUT CARD-FILE
      OUTPUT DISK-FILE
MOVE "NO" TO END-OF-FILE
PERFORM COPY-A-RECORD
      UNTIL END-OF-FILE EQUAL "YES"
CLOSE CARD-FILE
      DISK-FILE
STOP RUN

```

COPY-A-RECORD.

```

READ CARD-FILE
      AT END MOVE "YES" TO END-OF-FILE

IF END-OF-FILE EQUAL "NO"
      MOVE CARD-RECORD TO DISK-RECORD
      WRITE DISK-RECORD

```

٦ - ٩٦. بافتراض أن كل الملفات فتحت ... إلخ ، حدد الخطأ فيما يلي:

```

WRITE SAMPLE-OUTPUT-RECORD
      FROM WS-OUTPUT-RECORD-AREA

```

```

COMMENT -- DISPLAY RECORD JUST WRITTEN:
*      FOR DEBUGGING

```

```

DISPLAY SAMPLE-OUTPUT-RECORD

```

بعد تنفيذ عبارة WRITE .. لا يشير وصف السجل المسمى SAMPLE- OUTPUT- RECORD للسجل المنطقي الذي كتب توا . وبدلاً من ذلك .. فهو يشير الى موقع سجل منطقي جديد في ذاكرة احتياطية للملف ؛ يمكن استخدامها في عمل السجل المنطقي التالي المراد كتابته . وتعرض عبارة DISPLAY نفايا ، ولاتعرض السجل الذي كتب توا .

٦ - ٩٧ صحح الخطأ المذكور في المسألة السابقة ؛ حيث ان جزء WRITE... FROM قد استخدم ، فيكون التصحيح التعويض به :

```

DISPLAY WS-OUTPUT-RECORD-AREA

```

(بعد التنفيذ لعبارة WRITE .. فإن WS- OUTPUT- RECORD- AREA يظل محتويًا على السجل الذي كتب توا) .

إذا كانت في المسألة السابقة عبارة WRITE بسيطة ؛ أي فيها :

WRITE SAMPLE-OUTPUT-RECORD
DISPLAY SAMPLE-OUTPUT-RECORD

فيكون التصحيح سهلاً، ويأخذ الشكل التالي :

DISPLAY SAMPLE-OUTPUT-RECORD
WRITE SAMPLE-OUTPUT-RECORD

٦ - ٩٨ حدد الخطأ فيما يلي :

FD INPUT-FILE
RECORD CONTAINS 50 TO 200 CHARACTERS
.....
OPEN INPUT INPUT-FILE
READ INPUT-FILE
INTO WS-INPUT-RECORD-AREA
AT END...

لاستخدم READ... INTO مع سجلات متغيرة الطول (انظر ملحق ح ؛ لمعرفة الموقف في كويل 1985 النمطي) .

٦ - ٩٩ بين كيف يصحح الخطأ الموجود في المسألة السابقة .

FD INPUT-FILE
RECORD CONTAINS 50 TO 200 CHARACTERS
.....
OPEN INPUT INPUT-FILE
READ INPUT-FILE
AT END...
MOVE INPUT-RECORD TO WS-INPUT-RECORD-AREA

استخدم عبارة READ تتبعها عبارة MOVE ، بدلا من READ... INTO ، وذلك مع السجلات متغيرة الطول . ومع نفس السجلات .. استخدمت كذلك عبارة MOVE .. يتبعها عبارة WRITE بدلا من استخدام WRITE... FROM .

٦ - ١٠٠ حدد الخطأ فيما يلي :

READ SAMPLE-INPUT-FILE
AT END
MOVE "YES" TO END-OF-FILE
WRITE REPORT-LINE
FROM WS-TOTAL-LINE
AT END-OF-PAGE
ADD 1 TO NUMBER-OF-PAGES

يجب أن تكون كل العبارات في جزء AT END أمرية ، وهنا WRITE... AT END- OF- PAGE هي شرطية ، وينتج عن ذلك خطأ تكويني .

٦ - ١.٠١ صحح الخطأ الموجود في المسألة السابقة .

إذا كان ضروريا تنفيذ عبارة شرطية، عندما تكون هناك قواعد تطلب وجود عبارة أمرية ، ... ضع العبارة الشرطية في مقطع، ثم استخدم عبارة PERFORM (وهي أمرية) في تنفيذ المقطع .

```
READ SAMPLE-INPUT-FILE
  AT END
    PERFORM AT-END-ROUTINE
```

.....
AT-END-ROUTINE.

```
MOVE "YES" TO END-OF-FILE
WRITE REPORT-LINE
  FROM WS-TOTAL-LINE
  AT END OF PAGE
  ADD 1 TO NUMBER-OF-PAGES
```

٦ - ١.٠٢ مطلوب إنتاج قائمة من مجموعة بطاقات ؛ حيث تطبع نسخة طبق الأصل من كل بطاقة (الثمانين عموداً لها) على الورق. يجب أن تعيد كل صفحة منطقية إنتاج 10 بطاقات بالضبط ، ويجب أن يكون فيها سطر عنوان، يحتوى على عنوان ورقم الصفحة والتاريخ في صورة mmyydd ، وفي صورة رقم اليوم من السنة، ووقت تنفيذ البرنامج على الكمبيوتر، كما يجب أن يكون للصفحة نهاية، بها الرسالة SO FAR- CARDS WERE PRINTED . علق على كيفية تحقيق البرنامج الموجود في شكل (٦ - ١٣) لهذا النشاط .

```
00001 IDENTIFICATION DIVISION.
00002 PROGRAM-ID. CARDLIST.
00003 AUTHOR. LARRY NEWCOMER.
00004 INSTALLATION. PENN STATE UNIVERSITY--YORK CAMPUS.
00005 DATE-WRITTEN. MAY 1983.
00006 DATE-COMPILED. MAY 9,1983.
00007 SECURITY. NONE.
00008 * CARDLIST PRODUCES AN 80/80 LISTING OF CARD CONTENTS.
00009 * ITS PURPOSE HERE IS TO ILLUSTRATE VARIOUS FORMS OF
00010 * WRITE ... BEFORE/AFTER ADVANCING ... WITH THE LINAGE
00011 * CLAUSE.
00012 ENVIRONMENT DIVISION.
00013 CONFIGURATION SECTION.
00014 SOURCE-COMPUTER. IBM-370.
00015 OBJECT-COMPUTER. IBM-370.
00016 INPUT-OUTPUT SECTION.
00017 FILE-CONTROL.
00018 SELECT CARD-FILE ASSIGN TO CARDS.
00019 SELECT PRINT-FILE ASSIGN TO PRINTER.
00020 DATA DIVISION.
```

```

00021      FILE SECTION.
00022      FD  CARD-FILE
00023          RECORD CONTAINS 80 CHARACTERS
00024          LABEL RECORDS ARE OMITTED
00025
00026      01  CARD-INPUT                                PIC X(80).

00028      * -----
00029      FD  PRINT-FILE
00030          RECORD CONTAINS 132 CHARACTERS
00031          LABEL RECORDS ARE OMITTED
00032          * LINAGE IS 13
00033              WITH FOOTING AT 11
00034              LINES AT TOP 2
00035              LINES AT BOTTOM 2
00036
00037
00038      01  PRINT-LINE                                PIC X(132).
00039      * -----

00041      WORKING-STORAGE SECTION.
00042
00043      * -----
00044      01  WS-DATE-AND-TIME-AREAS.
00045          05  WS-REGULAR-DATE.
00046              10  WS-YY                                PIC 99.
00047              10  WS-MM                                PIC 99.
00048              10  WS-DD                                PIC 99.
00049      * -----
00050
00051      01  PROGRAM-SWITCHES-AND-COUNTERS.
00052          05  END-OF-CARDS-SWITCH                    PIC X(3).
00053          05  WS-PAGE-NUMBER                          PIC S9(3)      COMP-3.
00054          05  WS-CARD-COUNT                          PIC S9(3)      COMP-3.
00055      01  WS-LINE-AREA.
00056          05  WS-CARD-AREA                            PIC X(80).
00057          05  FILLER                                PIC X(52)      VALUE SPACES.
00058
00059      * -----
00060      01  WS-HEADING-AREA.
00061          05  FILLER                                PIC X(6)      VALUE "DATE ".
00062          05  HEADING-MM                            PIC Z9.
00063          05  FILLER                                PIC X      VALUE "/".
00064          05  HEADING-DD                            PIC 99.
00065          05  FILLER                                PIC X      VALUE "/".
00066          05  HEADING-YY                            PIC 99.
00067          05  FILLER                                PIC X(9)      VALUE ", JULIAN ".
00068          05  HEADING-JULIAN                        PIC Z9B999.
00069          05  FILLER                                PIC X(7)      VALUE ", TIME ".
00070          05  HEADING-TIME                          PIC 99B99B99B99.
00071          05  FILLER                                PIC X(6)      VALUE " PAGE.".
00072          05  WS-HEADING-PAGE-NUMBER                PIC ZZ9.
00073          05  FILLER                                PIC X(76)     VALUE SPACES.
00074      * -----
00075
00076      01  WS-FOOTING-AREA.
00077          05  FILLER                                PIC X(7)      VALUE "SO FAR ".
00078          05  WS-FOOTING-COUNT                      PIC ZZ9.
00079          05  FILLER                                PIC X(122)
00080              VALUE " CARDS WERE PRINTED".

00082      PROCEDURE DIVISION.

00084          MOVE "NO " TO END-OF-CARDS-SWITCH
00085          MOVE ZERO TO WS-PAGE-NUMBER
00086              WS-CARD-COUNT

```

```

00087
00088      ACCEPT WS-REGULAR-DATE      FROM DATE
00089      ACCEPT HEADING-JULIAN      FROM DAY
00090      ACCEPT HEADING-TIME      FROM TIME
00091
00092      MOVE WS-MM      TO HEADING-MM
00093      MOVE WS-DD      TO HEADING-DD
00094      MOVE WS-YY      TO HEADING-YY
00095
00096      OPEN      INPUT      CARD-FILE
00097      OUTPUT      PRINT-FILE
00098
00099      PERFORM INPUT-A-RECORD
00100
00101      PERFORM PRODUCE-PAGE-HEADING
00102
00103      PERFORM PRODUCE-80-80-LISTING
00104      UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00105
00106      CLOSE      CARD-FILE
00107      PRINT-FILE
00108
00109      STOP RUN
00109      .

00111      INPUT-A-RECORD.
00112
00113      READ CARD-FILE RECORD
00114      INTO WS-CARD-AREA
00115      AT END
00116      MOVE "YES" TO END-OF-CARDS-SWITCH
00117      .

00119      PRODUCE-80-80-LISTING.
00120
00121      ADD 1 TO WS-CARD-COUNT
00122
00123      WRITE PRINT-LINE
00124      FROM WS-LINE-AREA
00125      AFTER ADVANCING 1 LINE
00126      AT END-OF-PAGE
00127      PERFORM PRODUCE-PAGE-FOOTING
00128      PERFORM PRODUCE-PAGE-HEADING
00129      .
00130
00131      PERFORM INPUT-A-RECORD
00132      .

00134      PRODUCE-PAGE-HEADING.
00135
00136      ADD 1 TO WS-PAGE-NUMBER
00137      MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE-NUMBER
00138      WRITE PRINT-LINE
00139      FROM WS-HEADING-AREA
00140      AFTER ADVANCING PAGE
00141      .
00142
00143      PRODUCE-PAGE-FOOTING.
00144
00145      MOVE WS-CARD-COUNT TO WS-FOOTING-COUNT
00146
00147      WRITE PRINT-LINE
00148      FROM WS-FOOTING-AREA
00149      AFTER ADVANCING 2 LINES
00150      .

```


(١) كما يبدأ البرنامج بقراءة أول بطاقة (بعد فتح الملفات فوراً) .. فلا بد أن يبدأ بطباعة عنوان في بداية الصفحة الأولى . ويسمح وضع مقطع العنوان في هذا المقطع الخاص بتنفيذه في بداية جزء الإجراءات وتنفيذه مرة أخرى ، كلما كانت هناك حاجة لذلك ، في مقطع AT END- OF- PAGE من جزء الإجراءات . وحيث إن الطباعة تبدأ في بداية أول صفحة منطقية .. فلا تكتشف END- OF- PAGE ، في أول كتابة WRITE للملف . ويؤكد السطر الخامس من شكل (٦ - ١٣) أن أول صفحة سيكون بها عنوان .

(٢) يختبر النظام بالنسبة للسريان الزائد النهاية footing قبل اختباره للسريان الزائد للصفحة . وعندما تطبع البطاقة العاشرة (في السطر الحادي عشر بسبب طباعة سطر عنوان) يتميز السريان الزائد للنهاية ، وينفذ مقطع AT END- OF- PAGE ؛ متسبباً في طباعة سطر النهاية ، بعد ذلك .. يطبع سطر في بداية صفحة منطقية جديدة . وعلى هذا ، عندما تطبع البطاقة الحادية عشرة ، يوضع الورق فعلاً عند بداية الصفحة ؛ ولا يحدث سريان للصفحة . ونرى أن استخدام منطقة نهاية footing في نهاية الصفحة يمنع حدوث سريان زائد للصفحة؛ عن طريق كتابة سطر عنوان في بداية الصفحة التالية.

(٣) تتحدد قيمة LINAGE (السطر الثاني والثلاثون من شكل ٦ - ١٣) كما يلي : حيث إن العنوان يتطلب سطراً واحداً ، ونحن نريد عشرة سطور (تناظر عشرة بطاقات) في الصفحة ، إذ يجب أن تكون قيمة LINAGE مساوية 11 على الأقل (بدون نهاية footing الصفحة) . فإذا كان المطلوب طباعة النهاية بعد تقديم سطرين AFTER ADVANCING 2 LINES .. فإن السطرين الإضافيين يعتبران من حجم جسم الصفحة 13 . ويبين شكل ٦ - ١٤ عينة لصفحة مطبوعة .

```
DATE      5/09/83, JULIAN 83 129, TIME 13 08 12 41 PAGE    2
CARD      11
CARD      12
CARD      13
CARD      14
CARD      15
CARD      16
CARD      17
CARD      18
CARD      19
CARD      20
SO FAR    20 CARDS WERE PRINTED
```

شكل (٦ - ١٤)

(٤) الأسطر من ٨٨ إلى ٩٠ في شكل (٦ - ١٣) بها حقول منقحة HEADING- TIME , HEADING- JULIAN في مخزن العمل . يقبل التاريخ المعتاد في WS- REGULAR- DATE لأن النظام يحفظ التاريخ على هيئة yymmd (مذه هي الصيغة الأكثر راحة إذا كانت هناك حاجة للترتيب طبقاً للتاريخ ، والسنة التأثير الأكبر) . تنقل الحقول الجزئية day , month , year منفصلة إلى منطقة العنوان (انظر الأسطر من 92 إلى 94)؛ بحيث يمكن أن يعاد ترتيبها .

٦-١٠ البرنامج الموجود في شكل (٦-١٥)، هو نفسه مثل البرنامج الموجود في شكل (٦-١٣)، فيما عدى أن به WITH
 DEBUGGING MODE محددة في مقطع كمبيوتر المصدر، وأن عبارات DISPLAY مضافة إلى جزء الإجراءات؛
 للمساعدة في تصحيح البرنامج. ما مخرجات عبارات DISPLAY هذه؟

```

00084      PROCEDURE DIVISION.

00086      MOVE "NO " TO END-OF-CARDS-SWITCH
00087      MOVE ZERO TO WS-PAGE-NUMBER
00088      WS-CARD-COUNT
00089
00090      ACCEPT WS-REGULAR-DATE FROM DATE
00091      ACCEPT HEADING-JULIAN FROM DAY
00092      ACCEPT HEADING-TIME FROM TIME
00093
00094      D DISPLAY "STARTING VALUE OF END-OF-CARDS-SWITCH="
00095      D END-OF-CARDS-SWITCH
00096      D DISPLAY "STARTING VALUE OF WS-PAGE-NUMBER AND CARD-COUNT="
00097      D WS-PAGE-NUMBER WS-CARD-COUNT
00098      D DISPLAY "VALUE OF WS-REGULAR-DATE=" WS-REGULAR-DATE

00099      D DISPLAY "VALUE OF HEADING-TIME=" HEADING-TIME
00100      D DISPLAY "VALUE OF JULIAN DATE=" HEADING-JULIAN
00101
00102
00103      MOVE WS-MM TO HEADING-MM
00104      MOVE WS-DD TO HEADING-DD
00105      MOVE WS-YY TO HEADING-YY
00106
00107      OPEN INPUT CARD-FILE
00108      OUTPUT PRINT-FILE
00109
00110      PERFORM INPUT-A-RECORD
00111
00112      PERFORM PRODUCE-PAGE-HEADING
00113
00114      PERFORM PRODUCE-80-80-LISTING
00115      UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00116
00117      CLOSE CARD-FILE
00118      PRINT-FILE
00119      STOP RUN
00120

00122      INPUT-A-RECORD.

00123
00124      READ CARD-FILE RECORD
00125      INTO WS-CARD-AREA
00126      AT END
00127      MOVE "YES" TO END-OF-CARDS-SWITCH
00128
00129      D DISPLAY "JUST READ THE FOLLOWING CARD: " WS-CARD-AREA
00130      D DISPLAY "END-OF-CARDS-SWITCH IS: " END-OF-CARDS-SWITCH
00131      D
00132

00134      PRODUCE-80-80-LISTING.

00135
00136      ADD 1 TO WS-CARD-COUNT
00137
00138      WRITE PRINT-LINE
00139      FROM WS-LINE-AREA
00140      AFTER ADVANCING 1 LINE
00141      AT END-OF-PAGE
00142      PERFORM PRODUCE-PAGE-FOOTING
00143      PERFORM PRODUCE-PAGE-HEADING
    
```

```

00144
00145
00146          PERFORM INPUT-A-RECORD
00147

00149          PRODUCE-PAGE-HEADING.
00150
00151      D      DISPLAY "** ENTERING PRODUCE-PAGE-HEADING, CARD COUNT="
00152      D      WS-CARD-COUNT
00153
00154          ADD 1 TO WS-PAGE-NUMBER
00155          MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE-NUMBER
00156          WRITE PRINT-LINE
00157          FROM WS-HEADING-AREA
00158          AFTER ADVANCING PAGE
00159
00160
00161          PRODUCE-PAGE-FOOTING.
00162
00163      D      DISPLAY "** ENTERING PRODUCE-PAGE-FOOTING, CARD COUNT="
00164      D      WS-CARD-COUNT

00165
00166          MOVE WS-CARD-COUNT TO WS-FOOTING-COUNT
00167
00168          WRITE PRINT-LINE
00169          FROM WS-FOOTING-AREA
00170          AFTER ADVANCING 2 LINES
00171

```

شكل (٦ - ١٥)

انظر شكل (٦ - ١٦) لاحظ أن مترجم كويل IBM OS/VS يحذف إشارة العمليات من الحقول العددية المعروضة من عبارة DISPLAY إذا كانت هذه الحقول موجبة ، وعلى هذا يطبع WS-CARD-COUNT في صورة مقروءة . لاحظ كذلك أن اعداد COMP-3 تحولت إلى استخدام DISPLAY ؛ بفرض طباعتها بعبارة DISPLAY .

أدرس استخدام DISPLAY هذا والمخرجات المصاحبة له جيدا ؛ فهذا يمكن أن يوفر لك كثيرا من وقت التصحيح ، ولاحظ كذلك بصفة - خاصة كيف يعكس شكل (٦ - ١٦) ترتيب تنفيذ الكمبيوتر لعبارة البرنامج .

```

STARTING VALUE OF END-OF-CARDS-SWITCH=NO
STARTING VALUE OF WS-PAGE-NUMBER AND CARD-COUNT=000000
VALUE OF WS-REGULAR-DATE=830509
VALUE OF HEADING-TIME=13 14 00 72
VALUE OF JULIAN DATE=83 129
JUST READ THE FOLLOWING CARD: CARD 1
END-OF-CARDS-SWITCH IS: NO
** ENTERING PRODUCE-PAGE-HEADING, CARD COUNT=000
JUST READ THE FOLLOWING CARD: CARD 2
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 3
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 4
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 5
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 6
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 7
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 8
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 9
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 10

```

```

END-OF-CARDS-SWITCH IS: NO
** ENTERING PRODUCE-PAGE-FOOTING, CARD COUNT=010
** ENTERING PRODUCE-PAGE-HEADING, CARD COUNT=010
JUST READ THE FOLLOWING CARD: CARD 11
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 12
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 13
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 14
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD 15

```

شكل (٦ - ١٦)

٦ - ١٠٤ بين كيف تنتقل A والموصوف بالصورة PIC SV999 إلى B ، والذي له الصورة SV9 مع التقريب .

COMPUTE B ROUNDED = A

بالرغم من الشكل غير اللطيف فإن مايلي يؤدي نفس العمل .

ADD ZERO A GIVING B ROUNDED

٦ - ١٠٥ بين كيف يبدو الحقل المستقبل بعد كل عملية نقل تالية :

<i>Sending Field Value</i>	<i>Receiving Field PICTURE</i>
(a) 123.456	S9(5)V9(4)
(b) 123,456	S9(5)V9
(c) "STRING"	X(4) JUSTIFIED RIGHT
(d) "STRING"	X(4)
(e) "DOG"	X(5)
(f) "DOG"	X(5) JUSTIFIED RIGHT
(g) "DOG"	9(3)
(h) HIGH-VALUES	9(3)
(i) HIGH-VALUES	X(3)
(j) SPACES	9(3)
(k) ALL "\$"	X(5)
(l) 123.456	S99V99
(m) -123.456	999V999
(n) -123.456	\$\$\$\$.99-
(o) -123456	\$\$\$\$\$.99BCR

00123,4560 (a)

00123,4 (b)

RING (c)

STRI (d)

DOGbb (e)

bb DOG (f)

(g) غير صحيح.. فالنقل يخلط بين العددي وغير العددي .

(h) غير صحيح HIGH- VALUES غير عددي، وتنقل إلى حقل معرف بواسطة PIC X فقط .

(i) ٣ بايت لأعلى رمز في تسلسل التتابع .

(l) غير صحيح SPACES تنقل إلى عناصر حرفية أو حرفية عددية فقط

\$\$\$\$ (k)

(l) 23_^45 (يحدث حذف من ناحية اليسار، ومن ناحية اليمين) .(m) 123_^456 (تنتج القيمة المطلقة ، حيث يفتقر الحقل المستقبل إلى وجود إشارة) .

\$123.45- (n)

(o) \$23456.00bCR (يحذف الرقم الموجود على أقصى اليسار) .

١٠٦ - ٦ بمعرفة مايلي :

```

01 A.
05 B    PIC X(4)    VALUE "THIS".
05 C    PIC X(2)    VALUE "IS".
05 D    PIC X(3)    VALUE "FUN".

```

```

01 W.
05 X    PIC X(3).
05 Y    PIC X(2).
05 Z    PIC X(3).

```

اوجد محتويات: (ل) X (ب) Y (ج) Z بعد نقل A إلى W .

يعالج الكوبل التنقل لمجموعة عناصر، كما لو كان نقلا لحقل كبير من النوع الحرفي عددي . وهنا A يشغل ٩ بايت، وW يشغل ٨ بايت ، وعلى هذا يحذف البايت الموجود على أقصى اليمين من A (وهو البايت الموجود على أقصى اليمين من D) يحذف .

(a) "THI" (b) "SI" (c) "SFU"

١٠٧ - ٦ انقد مايلي :

```

01 A.
05 B    PIC S9(3)V99    COMP-3.
05 C    PIC S99          COMP.
05 D    PIC S99V9       DISPLAY.

```

MOVE ZEROS TO A

حيث إن A مجموعة عناصر فإنها تعامل كعنصر حرفي عددي له الاستخدام DISPLAY ، وتنقل أصفار الى حقول A. ويحدث هذا للعنصر D (حيث إن استخدامه من نوع DISPLAY) ، لكنه لا يحدث للعنصرين B , C . الطريقة الصحيحة هي:

MOVE ZEROS TO B C D

والتي تنقل أصفارا إلى كل حقل فردي ، مع التحويل التلقائي إلى الاستخدام USAGE الصحيح .

٦-١٠ انقد مايلى :

.....
STOP RUN
DISPLAY "ALL FILES ARE NOW CLOSED"

العبارات صحيحة من ناحية التكوين ، ولكن لا يمكن لعبارة DISPLAY أن تنفذ . عندما تنفذ عبارة STOP RUN .. يتوقف الكمبيوتر عن تنفيذ التعليمات من البرنامج الحالى .

٦-١٠ افترض أن أقصى أجر فى الساعة فى إحدى الشركات هو 55.00 دولار . افترض أن أقصى عدد ساعات عمل مسموح به فى الأسبوع 60 ساعة (مع دفع مرة ونصف من معدل الأجر المعتاد للساعات التى تزيد عن 40 ساعة) . إذا كان عدد العاملين بالشركة 1000 عامل ، اكتب PICTURE الخاصة بـ TOTAL- WEEKLY- PAYROLL بحيث لا يحدث خطأ فى الحجم .

إذا عمل أحد العاملين أقصى ساعات عمل، بأقصى معدل أجر ، فإن أجر العامل الأسبوعى سيكون

$$(40 \times 55.00) + (20 \times 55.00 \times 1.5) = 3850.00 \text{ dollars}$$

وعلى هذا .. لا يمكن أن يتعدى إجمالى الأجر الأسبوعى لجميع العاملين :

$$1000 \times 3850.00 = 3,850,000.00 \text{ dollars}$$

وعلى هذا .. فإن PIC S9(7)V99 تكون كافية .

٦-١١ عبر عن العلاقات الجبرية التالية باستخدام عبارة COMPUTE :

$$(a) \quad a = \frac{b}{c} + b - (c \times d) \quad (b) \quad a = \frac{b+c}{d} - \frac{c}{e+f} \quad (c) \quad d = \frac{(a+b)^2}{a-b} + a - (b+c)$$

- (a) COMPUTE A = B / C + B - C * D
(b) COMPUTE A = (B + C) / D - C / (E + F)
(c) COMPUTE D = (A + B) ** 2 / (A - B) + A - (B + C)


```

00033      01 MAILING-INPUT.
00034          05 MAILING-NAME          PIC X(20).
00035          05 MAILING-ADDRESS        PIC X(20).
00036          05 MAILING-CITY           PIC X(20).
00037          05 MAILING-STATE          PIC X(2).
00038          05 MAILING-ZIP            PIC X(5).
00039          05 FILLER                 PIC X(13).

00041      FD MAILING-LABEL-FILE
00042          RECORD CONTAINS 132 CHARACTERS
00043          LABEL RECORDS ARE OMITTED
00044          .
00045      01 LABEL-LINE                  PIC X(132).

00047      WORKING-STORAGE SECTION.
00048
00049      01 PROGRAM-SWITCHES-AND-COUNTERS.
00050          05 END-OF-FILE-SW          PIC X(3).
00051          05 WS-LABEL-COUNT          PIC S9(3).      COMP-3.
00052
00053      01 WS-NAME-LINE.
00054          05 NAME-1                  PIC X(30)      VALUE ALL "*".
00055          05 FILLER                  PIC X(10)      VALUE SPACES.
00056          05 NAME-2                  PIC X(30)      VALUE ALL "*".
00057          05 FILLER                  PIC X(62)      VALUE SPACES.
00058
00059      01 WS-ADDRESS-LINE.
00060          05 ADDRESS-1               PIC X(30)      VALUE ALL "*".
00061          05 FILLER                  PIC X(10)      VALUE SPACES.
00062          05 ADDRESS-2               PIC X(30)      VALUE ALL "*".
00063          05 FILLER                  PIC X(62)      VALUE SPACES.
00064
00065      01 WS-CITY-STATE-LINE.
00066          05 CITY-1                  PIC X(20)      VALUE ALL "*".
00067          05 FILLER                  PIC X          VALUE SPACES.
00068          05 STATE-1                PIC XX         VALUE ALL "*".
00069          05 FILLER                  PIC X(2)       VALUE SPACES.
00070          05 ZIP-1                   PIC X(5)       VALUE ALL "*".
00071          05 FILLER                  PIC X(10)      VALUE SPACES.
00072          05 CITY-2                 PIC X(20)      VALUE ALL "*".
00073          05 FILLER                  PIC X          VALUE SPACES.
00074          05 STATE-2                 PIC XX         VALUE ALL "*".
00075          05 FILLER                  PIC X(2)       VALUE SPACES.
00076          05 ZIP-2                   PIC X(5)       VALUE ALL "*".
00077          05 FILLER                  PIC X(62)      VALUE SPACES.

00079      01 WS-COUNT-LINE.
00080          05 PRINTED-COUNT            PIC Z,ZZ9.
00081          05 FILLER                  PIC X(15)
00082          VALUE " LABELS PRINTED".
00083          05 FILLER                  PIC X(112)      VALUE SPACES.

00085      PROCEDURE DIVISION.

00087          MOVE "NO " TO END-OF-FILE-SW
00088          MOVE ZERO TO WS-LABEL-COUNT
00089
00090          OPEN      INPUT          MAILING-MASTER
00091                  OUTPUT         MAILING-LABEL-FILE

```

شكل (٦ - ١٧) تكملة


```

00092
00093      PERFORM 030-PRINT-2-LABELS
00094      2 TIMES
00095
00096      PERFORM 020-PRODUCE-MAILING-LABELS
00097      UNTIL END-OF-FILE-SW IS EQUAL TO "YES"
00098
00099      PERFORM 040-PRODUCE-COUNT-LINE
00100
00101      CLOSE   MAILING-MASTER
00102      MAILING-LABEL-FILE
00103      STOP RUN
00104      .

00106      010-INPUT-2-RECORDS.
00107
00108      MOVE SPACES TO WS-NAME-LINE
00109      WS-ADDRESS-LINE
00110      WS-CITY-STATE-LINE
00111
00112      READ MAILING-MASTER RECORD
00113      AT END
00114      MOVE "YES" TO END-OF-FILE-SW
00115      .
00116      IF END-OF-FILE-SW = "NO"
00117
00118      ADD 1 TO WS-LABEL-COUNT
00119      MOVE MAILING-NAME      TO NAME-1
00120      MOVE MAILING-ADDRESS  TO ADDRESS-1
00121      MOVE MAILING-CITY     TO CITY-1
00122      MOVE MAILING-STATE    TO STATE-1
00123      MOVE MAILING-ZIP      TO ZIP-1
00124
00125      READ MAILING-MASTER RECORD
00126      AT END
00127      MOVE "YES" TO END-OF-FILE-SW
00128      .
00129      IF END-OF-FILE-SW EQUAL "NO"
00130
00131      ADD 1 TO WS-LABEL-COUNT
00132      MOVE MAILING-NAME      TO NAME-2
00133      MOVE MAILING-ADDRESS  TO ADDRESS-2
00134      MOVE MAILING-CITY     TO CITY-2
00135      MOVE MAILING-STATE    TO STATE-2
00136      MOVE MAILING-ZIP      TO ZIP-2
00137      .

00139      020-PRODUCE-MAILING-LABELS.
00140
00141      PERFORM 010-INPUT-2-RECORDS
00142
00143      PERFORM 030-PRINT-2-LABELS
00144      .
00145
00146      030-PRINT-2-LABELS.
00147
00148      WRITE LABEL-LINE
00149      FROM WS-NAME-LINE
00150      BEFORE ADVANCING 1 LINE
00151      WRITE LABEL-LINE

```

شكل (٦ - ١٧) تكملة

```

00152          FROM WS-ADDRESS-LINE
00153          BEFORE ADVANCING 1 LINE
00154      WRITE LABEL-LINE
00155          FROM WS-CITY-STATE-LINE
00156          BEFORE ADVANCING 2 LINES
00157
00158
00159      040-PRODUCE-COUNT-LINE.
00160
00161          MOVE WS-LABEL-COUNT TO PRINTED-COUNT
00162      WRITE LABEL-LINE
00163          FROM WS-COUNT-LINE
00164          AFTER ADVANCING 2 LINES
00165

```

شكل (٦ - ١٧) تكملة

NATHAN CONVERSE 709 N. SIMPSON ST. MIAMI	FL 10022	MIKE PERELMAN 213 E. SOUTH ST. HANOVER	MD 23451
BARNEY PERELMAN 782 N. SOUTH ST. NEW YORK	NY 10012	KATHY BUCHER 666 W. EAST ST. PHILADELPHIA	PA 34256
ABE SHARP 783 S. NORTH ST. BALTIMORE	MD 22145		

5 LABELS PRINTED

شكل (٦ - ١٨)

٦ - ١١٣ اكتب برنامجاً يحسب إحصائيات مبيعات مبنية على عينة من المبيعات الشهرية لأحد المحلات متعدد المخازن . تحتوي المدخلات على سجلات منطقية تشمل مايلي : معرفاً للمخزن، ورمزاً للموقع، وكمية المبيعات الشهرية . وتحتوي المخرجات على تقرير مطبوع يبين سطرًا تفصيليًا، يشمل معلومات عن كل مخزن ، يتبعه سطر نهائى يعطى ، للعينة ، متوسط المبيعات والانحراف المعيارى للمبيعات .

تحسب كميات العينة من العلاقة التالية :

$$\begin{aligned}
 \text{sample total} &= s_1 + s_2 + \dots + s_n \\
 \text{average} &= \frac{\text{sample total}}{n} = \frac{s_1 + s_2 + \dots + s_n}{n} \\
 \text{standard deviation} &= \sqrt{\frac{n(s_1^2 + s_2^2 + \dots + s_n^2) - (s_1 + s_2 + \dots + s_n)^2}{n(n-1)}}
 \end{aligned}$$

حيث n عدد المخازن المأخوذة فى العينة، و s_i كمية مبيعات المخزن رقم i .

الإجابة : انظر شكل (٦ - ١٩) (توجد مخرجات تقليدية فى شكل ٦ - ٢٠) . بالرغم من أنه غير مطلوب ، إلا أن هذا البرنامج يطبع سطر نهاية يعطى إجمالاً جاريًا للمبيعات .

```

00001      IDENTIFICATION DIVISION.
00002
00003      PROGRAM-ID. SALESTAT.
00004      AUTHOR. LARRY NEWCOMER.
00005      INSTALLATION. PENN STATE UNIVERSITY--YORK CAMPUS.
00006      DATE-WRITTEN. MAY 1983.
00007      DATE-COMPILED. MAY 9,1983.
00008      SECURITY. NONE.
00009
00010      * SALESTAT PRODUCES A LISTING OF MONTHLY SALES BY RETAIL
00011      * STORE. IT PRINTS A FINAL LINE SHOWING THE TOTAL SALES,
00012      * THE AVERAGE SALES, AND THE STANDARD DEVIATION FOR SALES.
00013
00014      * SALESTAT PROVIDES A GOOD ILLUSTRATION OF THE USE OF
00015      * THE COMPUTE STATEMENT.
00016
00017      ENVIRONMENT DIVISION.
00018
00019      CONFIGURATION SECTION.
00020      SOURCE-COMPUTER. IBM-370.
00021      OBJECT-COMPUTER. IBM-370.
00022      INPUT-OUTPUT SECTION.
00023      FILE-CONTROL.
00024          SELECT SALES-FILE              ASSIGN TO SALES.
00025          SELECT SALES-REPORT            ASSIGN TO SALELIST.
00026
00027      DATA DIVISION.
00028
00029      FILE SECTION.
00030
00031      FD SALES-FILE
00032          RECORD CONTAINS 80 CHARACTERS
00033          LABEL RECORDS ARE OMITTED
00034      .
00035      01 SALES-INPUT                      PIC X(80).

00037      FD SALES-REPORT
00038          RECORD CONTAINS 132 CHARACTERS
00039          LABEL RECORDS ARE OMITTED
00040          LINAGE IS 10
00041              WITH FOOTING AT 8
00042              LINES AT TOP 2
00043              LINES AT BOTTOM 2
00044
00045
00046      01 SALES-LINE                      PIC X(132).

00048      WORKING-STORAGE SECTION.
00049
00050      01 WS-DATE-AND-TIME-AREAS.
00051          05 WS-REGULAR-DATE.
00052              10 WS-YY                  PIC 99.
00053              10 WS-MM                  PIC 99.
    
```

```

00054          10 WS-DD                      PIC 99.
00055
00056      01 PROGRAM-SWITCHES-COUNTERS-SUMS.
00057          05 END-OF-SALES-SWITCH          PIC X(3).
00058          05 WS-PAGE-NUMBER                PIC S9(3)      COMP-3.
00059          05 WS-TOTAL-SALES                PIC S9(7)V99    COMP-3.
00060          05 WS-NUMBER-STORES               PIC S9(5)      COMP-3.
00061          05 WS-SUM-SALES-SQUARED          PIC S9(15)V99   COMP-3.
00062
00063      01 WS-SALES-AREA.
00064          05 WS-SALES-STORE-ID              PIC X(5).
00065          05 WS-SALES-LOCATION-CODE          PIC XX.
00066          05 WS-SALES-SALES                PIC S9(6)V99.
00067          05 FILLER                        PIC X(65).
00068
00069      01 WS-LINE-AREA.
00070          05 WS-LINE-STORE-ID              PIC X(5).
00071          05 FILLER                        PIC X(5)      VALUE SPACES.
00072          05 WS-LINE-LOCATION-CODE          PIC XX.
00073          05 FILLER                        PIC X(5)      VALUE SPACES.
00074          05 WS-LINE-SALES                PIC ZZZ,ZZZ.99.
00075          05 FILLER                        PIC X(105)     VALUE SPACES.
00076
00077      01 WS-HEADING-AREA-1.
00078          05 FILLER                        PIC X(1)      VALUE SPACES.
00079          05 FILLER                        PIC X(20)
00080          VALUE "SALES STATISTICS".
00081          05 HEADING-MM                    PIC Z9.
00082          05 FILLER                        PIC X          VALUE "/".
00083          05 HEADING-DD                    PIC 99.
00084          05 FILLER                        PIC X          VALUE "/".
00085          05 HEADING-YY                    PIC 99.
00086          05 FILLER                        PIC X(4)      VALUE SPACES.
00087          05 FILLER                        PIC X(6)      VALUE "PAGE ".
00088          05 WS-HEADING-PAGE-NUMBER        PIC ZZ9.
00089          05 FILLER                        PIC X(90)     VALUE SPACES.
00090
00091      01 WS-HEADING-AREA-2.
00092          05 FILLER                        PIC X(8)      VALUE "STORE ID".
00093          05 FILLER                        PIC X(2)      VALUE SPACES.
00094          05 FILLER                        PIC X(12)     VALUE "LOC".
00095          05 FILLER                        PIC X(10)     VALUE "SALES".
00096          05 FILLER                        PIC X(100)    VALUE SPACES.
00097
00098      01 WS-FOOTING-AREA.
00099          05 FILLER                        PIC X(15)     VALUE SPACES.
00100          05 WS-FOOTING-TOTAL              PIC Z,ZZZ,ZZZ.99.
00101          05 FILLER                        PIC X(105)    VALUE " *".
00102
00103      01 WS-FINAL-LINE-1.
00104          05 FILLER                        PIC X(15)     VALUE SPACES.
00105          05 WS-FINAL-TOTAL                PIC Z,ZZZ,ZZZ.99.
00106          05 FILLER                        PIC X(105)    VALUE " **".
00107
00108      01 WS-FINAL-LINE-2.
00109          05 FILLER                        PIC X(10)     VALUE "AVERAGE--".
00110          05 WS-AVERAGE-SALES              PIC ZZZ,ZZZ.99.
00111          05 FILLER                        PIC X(6)      VALUE " SD-- ".
00112          05 WS-SD                          PIC ZZZ,ZZZ.99.
00113          05 FILLER                        PIC X(96)     VALUE SPACES.
00114
00115      PROCEDURE DIVISION.
00116
00117          MOVE "NO " TO END-OF-SALES-SWITCH
00118          MOVE ZERO TO WS-PAGE-NUMBER
00119          WS-TOTAL-SALES

```

```

00120                                WS-NUMBER-STOES
00121                                WS-SUM-SALES-SQUARED
00122
00123                                ACCEPT WS-REGULAR-DATE      FROM DATE
00124
00125                                MOVE WS-MM                  TO HEADING-MM
00126                                MOVE WS-DD                  TO HEADING-DD
00127                                MOVE WS-YY                  TO HEADING-YY
00128
00129                                OPEN      INPUT              SALES-FILE
00130                                OUTPUT              SALES-REPORT
00131
00132                                PERFORM GET-SALES-RECORD
00133
00134                                PERFORM PRODUCE-PAGE-HEADING
00135
00136                                PERFORM PRODUCE-SALES-LINE
00137                                UNTIL END-OF-SALES-SWITCH IS EQUAL TO "YES"
00138
00139                                PERFORM PRODUCE-FINAL-LINES
00140
00141                                CLOSE    SALES-FILE
00142                                SALES-REPORT
00143                                STOP RUN
00144                                .

00146                                GET-SALES-RECORD.
00147
00148                                READ SALES-FILE RECORD
00149                                INTO WS-SALES-AREA
00150                                AT END
00151                                MOVE "YES" TO END-OF-SALES-SWITCH
00152                                .
00153

00155                                PRODUCE-SALES-LINE.
00156
00157                                ADD 1 TO WS-NUMBER-STOES
00158                                COMPUTE WS-SUM-SALES-SQUARED = WS-SUM-SALES-SQUARED
00159                                                                + WS-SALES-SALES ** 2
00160
00161                                ADD WS-SALES-SALES TO WS-TOTAL-SALES
00162
00163                                MOVE WS-SALES-STORE-ID      TO WS-LINE-STORE-ID
00164                                MOVE WS-SALES-LOCATION-CODE    TO WS-LINE-LOCATION-CODE
00165                                MOVE WS-SALES-SALES          TO WS-LINE-SALES
00166
00167                                WRITE SALES-LINE
00168                                FROM WS-LINE-AREA
00169                                AFTER ADVANCING 2 LINES
00170                                AT END-OF-PAGE
00171                                PERFORM PRODUCE-PAGE-FOOTING
00172                                PERFORM PRODUCE-PAGE-HEADING
00173                                .
00174
00175                                PERFORM GET-SALES-RECORD
00176                                .

00178                                PRODUCE-PAGE-HEADING.
00179

```

```

00180      ADD 1 TO WS-PAGE-NUMBER
00181      MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE-NUMBER
00182      WRITE SALES-LINE
00183          FROM WS-HEADING-AREA-1
00184          AFTER ADVANCING PAGE
00185      WRITE SALES-LINE

00186          FROM WS-HEADING-AREA-2
00187          AFTER ADVANCING 1 LINE
00188
00189
00190      PRODUCE-PAGE-FOOTING.
00191
00192          MOVE WS-TOTAL-SALES TO WS-FOOTING-TOTAL
00193
00194      WRITE SALES-LINE
00195          FROM WS-FOOTING-AREA
00196          AFTER ADVANCING 2 LINES
00197
00198
00199      PRODUCE-FINAL-LINES.
00200
00201          MOVE WS-TOTAL-SALES          TO WS-FINAL-TOTAL
00202      WRITE SALES-LINE
00203          FROM WS-FINAL-LINE-1
00204          AFTER ADVANCING 2 LINES
00205
00206      COMPUTE WS-AVERAGE-SALES = WS-TOTAL-SALES / WS-NUMBER-STORES
00207
00208      COMPUTE WS-SD = ((WS-NUMBER-STORES * WS-SUM-SALES-SQUARED
00209                      - WS-TOTAL-SALES ** 2)
00210                      /
00211                      (WS-NUMBER-STORES * (WS-NUMBER-STORES - 1))) ** .5
00212
00213      WRITE SALES-LINE
00214          FROM WS-FINAL-LINE-2
00215          AFTER ADVANCING 2 LINES
00216

```

شكل (٦ - ١٩) تكملة

SALES STATISTICS			5/09/83	PAGE	1
STORE ID	LOC	SALES			
00001	AA	1,000.00			
00002	BB	2,000.00			
00003	CC	3,000.00			
		6,000.00 *			
SALES STATISTICS			5/09/83	PAGE	2
STORE ID	LOC	SALES			
00004	DD	4,000.00			
00005	EE	5,000.00			
00006	FF	6,000.00			
		21,000.00 *			
SALES STATISTICS			5/09/83	PAGE	3
STORE ID	LOC	SALES			
00007	GG	7,000.00			
00008	HH	8,000.00			
		36,000.00 **			
AVERAGE--	4,500.00	SD--	2,449.49		

شكل (٦ - ٢٠)

نماذج برهجة

٦ - ١١٤ بمعرفة الأساس P المستثمر في حساب إخبار بمعدل فائدة I بحسب بفائدة مركبة عدد C من المرات خلال السنة ، فإن إجمالي المبلغ المتراكم T خلال N سنة هو :

$$T = P * \left(1 + \frac{I}{C}\right)^{C * N}$$

اكتب برنامجا بلغة الكويل يقوم بإدخال C, I, P، ويطلع تقريراً مبيناً T, N, C, I, P. اطلع عدداً من الأسطر يناظر عدد سطور المدخلات. صمم سجلات المدخلات وتخطيط التقرير، واستخدم جزء LINAGE: للتأكد من أن (١) وجود ٢٠ سطر تفصيلياً مع مسافة مزدوجة بينها في كل صفحة منطقية، (٢) وجود هامش علوي وهامش سفلي بعرض ثلاثة سطور لكل منها (٣) عنوان وعناوين أعمدة في بداية جسم كل صفحة (بما في ذلك رقم الصفحة وتاريخ وقت التنفيذ).

٦ - ١١٥ اكتب برنامجاً بلغة الكويل، يدخل سجلات مخزون، طول السجل ٨٠ بايت، وله الشكل التالي :

العمدة ١ - ٦ : رقم العنصر

العمدة من ٧ - ٢٦ : وصف العنصر

الاعدة من ٢٧ - ٣١ : الكمية الموجودة

الاعدة من ٣٢ - ٣٨ : التكلفة (مع ترك خانتين للكسر العشري)

البقية : غير مستخدم

اطبع تقريراً به سطر لكل عنصر، مبيناً مايلي :

item #	description	quantity on hand	cost	worth
-----------	-------------	---------------------	------	-------

حيث worth هي حاصل ضرب الكمية الموجودة quantity on hand في التكلفة cost . أطلع في نهاية التقرير سطرًا نهائيًا يبين: (١) إجمالي قيمة worth المخزون (أي مجموع القيمة لكل العناصر) (٢) ومتوسط تكلفة العنصر (خارج قسمة إجمالي المخزون على مجموع الكمية الموجودة) .

صمم الصفحة المنطقية بنفسك على أن تشمل: (١) عناوين، (٢) منطقة نهاية تظهر فيها القيمة الحالية لإجمالي الكمية الموجودة (انظر شكل (٦ - ٢٠)، (٣) رقمًا للصفحة وتاريخ وقت التنفيذ في بداية كل صفحة .

٦ - ١١٦ لدى إحدى المكاتب الاستشارية معدل داخلي internal rate لكل العاملين فيها (مثل هذا المعدل مايدفعه المكتب لكل عامل في الساعة)، ولديها معدل خارجي external rate (مايدفعه العميل لكل ساعة عمل للعامل بالمكتب) . بمعرفة المدخلات التالية :

الاعدة من ١ - ٤ : رقم تعريف العامل

الاعدة من ٥ - ٩ : المعدل الداخلي (خانتين للكسر العشري)

الاعدة من ١٠ - ١٤ : المعدل الخارجي (خانتين للكسر)

الاعدة من ١٥ - ١٧ : عدد ساعات العمل الاسبوعية (خانة للكسر العشري)

اطبع تقريراً للرواتب يبين :

employee ID	hours worked	internal rate	payroll amount	external rate	amount charged	gross profit
----------------	-----------------	------------------	-------------------	------------------	-------------------	-----------------

حيث

$$\text{payroll amount} = \text{hours} * \text{internal rate}$$

$$\text{amount charged} = \text{hours} * \text{external rate}$$

$$\text{gross profit} = \text{amount charged} - \text{payroll amount}$$

وفي نهاية التقرير ، اطلع سطرًا يبين إجمالي قيمة الرواتب ، وإجمالي القيمة المطلوبة من العملاء ، ومتوسط إجمالي الربح لكل عامل بالمكتب . يجب أن تحتوى الصفحة المنطقية على مايلي :

(١) عناوين (٢) نهاية تبين إجمالي عدد العاملين الذي أجرى لهم تشغيل حتى الآن (٣) رقم الصفحة وتاريخ وقت تنفيذ البرنامج في بداية كل صفحة .

الفصل السابع

منطق البرنامج

Program Logic

٧ - ١ التصميم المنطقي : خرائط المسار المرتبة

يشير منطق البرنامج program logic الى ترتيب تنفيذ عبارات البرنامج (على عكس ترتيبها الواقعي أو الطبيعي الموجوده فيه في قائمة المصدر) . خريطة المسار المرتبة structured flowchart هي رسم يحدد ترتيب العمليات في الخوارزمي . وعلى هذا .. يمكن استخدام خريطة مسار مرتبة في تصميم منطق جزء حل المشكلة من البرنامج ، قبل أن يكتب البرنامج فعلا بالكوبل (أو بأى لغة أخرى) . إذا ما حفظت خريطة المسار مجددة ، تعكس التغييرات في البرنامج ، فإنها تخدم كذلك كتوثيق مفيد للبرنامج .

والرموز النمطية لخريطة المسار موضحة في شكل ٧ . ١ .

٧ - ٢ هياكل منطق البرنامج

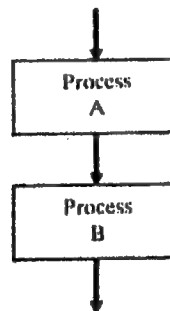
يمكن رؤية أن منطق أى خوارزمي أو برنامج كمبيوتر يمكن تحقيقه كخليط من ثلاثة هياكل منطقية logic structures أساسية .

هيكل التتابع (مكون التتابع)

هذا هو تنفيذ تتابعى لعمليات أو تعليمات ، واحدة تلو الأخرى . وفي برنامج الكمبيوتر المحتوى على هيكل تتابع .. يتفق الترتيب المنطقي للعبارات مع ترتيبها الطبيعي . ويبين شكل ٧ . ٢ خريطة مسار لهيكل التتابع .

رمز النهاية أو البداية ، يحمل كلمة START (أو STOP أو END) داخله ، يعرف بداية (أو نهاية) الخوارزمي . يمكن كتابة اسم الخوارزمي داخله كذلك .	
رمز العملية ، يحدد تشغيل الكمبيوتر لمعلومات . توصف طبيعة التشغيل داخل الرمز .	
رمز القرار ، يحدد قرار الكمبيوتر للاختيار بين مسارين منطقيين في الخوارزمي . ويتميز بسنهمين تاركين الرمز ، مبينا البديلين من المسارين المنطقيين .	
رمز العملية سبق تعريفها ، ويمثل تشغيل كمبيوتر معقد بدرجة كافية ليتطلب خريطة مسار منفصلة تصفه . يحتوي الرمز على اسم المقطع المعد له خريطة مسار منفصلة .	
رمز الواصل ، يمثل توصيلا لجزئين أو أكثر من خريطة المسار .	
رمز منخالات أو مخرجات ، يحدد عملية منخلات وعملية مخرجات .	
أسهم تستخدم في توصيل الرموز السابقة في الترتيب التي تنفذ به .	

شكل (٧ - ١)



شكل (٧ - ٢)

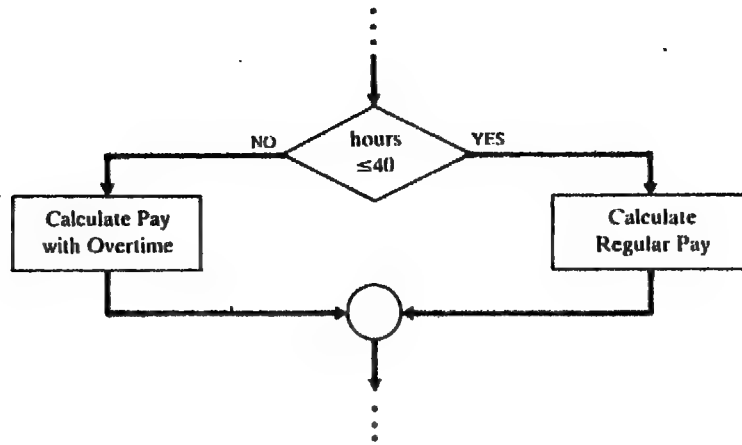
هيكل الاختيار (مكون الاختيار)

يختار الهيكل أحد فرعين اثنين بالضبط (لعمليات أو مخرجات من التعليمات) . ويعتمد اختيار الفرع على القيمة الحالية لشرط ما والتي تكون صحيح true أو خطأ false دائما .

مثال ٧ - ١ :

في برنامج الرواتب .. تكون هناك حاجة لإحدى مجموعات التعليمات لحساب إجمالي راتب العامل الذي يعمل 40 ساعة أو أقل ، ومجموعة أخرى من التعليمات للعامل الذي يعمل وقتا إضافيا . وهيكل الاختيار مبين في شكل ٧ . ٣ .

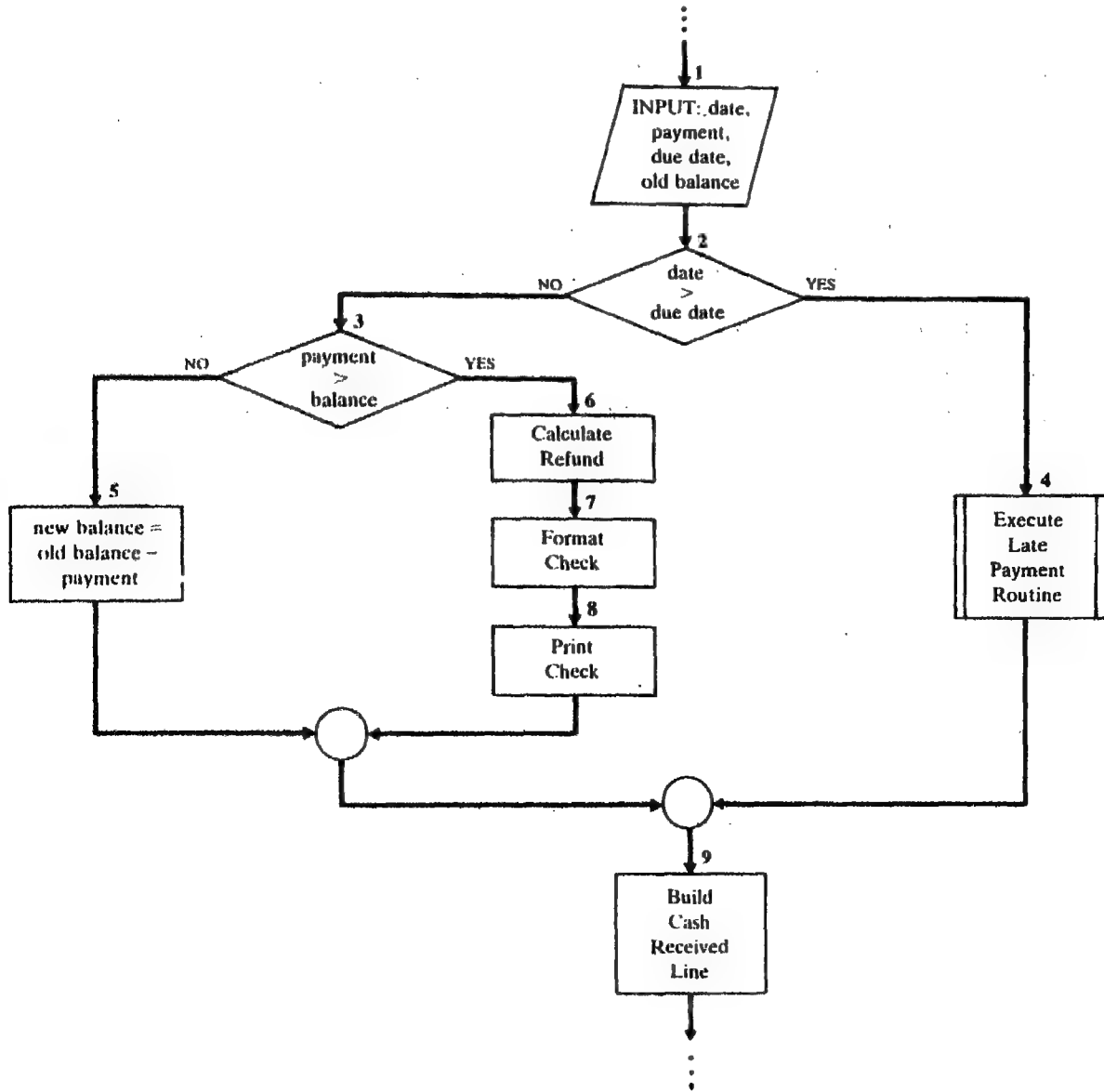
يجب أن يحتوى رمز القرار على شرط ، إما أن تكون نتيجته صحيح true أو خطأ false . وكما في الحالة الحالية .. فإنه عادة ما يشمل الشرط عنصرى بيانات اثنين (أو عنصر بيانات وثابت) لتحديد ما إذا تحققت علاقة معينة (أقل من أو يساوى هنا) أم لا . فإذا كان الشرط صحيحا ، فيتبع السهم الخاص بكلمة YES أو TRUE في تنفيذ هيكل الاختيار ، وإلا فيتبع السهم الخاص بكلمة NO أو FALSE .



شكل (٧ - ٣)

مثال ٧ - ٢ :

تدمج خريطة المسار المرتبة الموجودة في شكل ٧ . ٤ هيكل التتابع مع هيكل الاختيار . الرمز 1 أو الرمز 2 والرمز 9 تمثل تتابع ، الجزء الثانى ، والذى منه الرمز 2 ، هو بنفسه هيكل اختيار (بمسارين منطقيين بديلين خلال الرمز 3 و 4 الرمز 4 هو عملية سابقة التعريف) ، تمثل مجموعة (من الممكن أن تكون معقدة) من الإجراءات المعرفة في خريطة مسار أخرى . الرمز 3 هو رأس هيكل اختيار آخر ، بمسارين منطقيين بديلين خلال الرمز 5 من ناحية ، 6 و 7 و 8 من ناحية أخرى . يلاحظ أن الرموز 6 و 7 و 8 تكون هيكل تتابع بنفسها .



شكل (٧ - ٤)

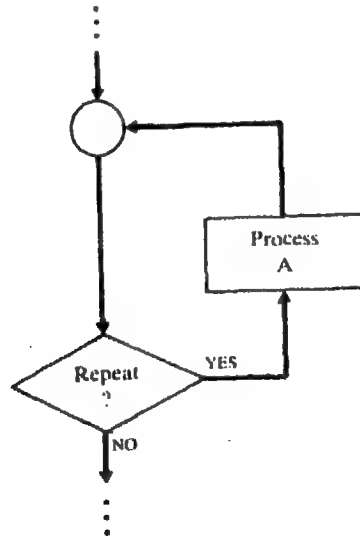
هيكل التكرار (مكون التكرار)

يستخدم هيكل التكرار عندما يجب تكرار عملية أو مجموعة من العمليات لعدد معين من المرات، وهو هيكل دورة البرنامج . program loop

مثال ٧ - ٣ :

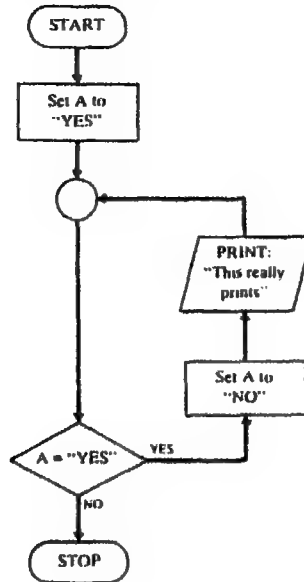
صيغة هيكل التكرار مرسوم لها خريطة مسار في شكل ٧ . ٥ ، وعادة ما تسمى هذه الصيغة بهيكل DO WHILE . عندما يدخل الهيكل التنفيذ .. فإن أول شيء ما يحدث هو اتخاذ قرار ماذا كان سيحدث تكرار أما لا .

إذا كان القرار لا (شرط التكرار غير صحيح) ، فيتم الخروج من الهيكل بون تنفيذ العملية A . إذا كان القرار نعم (شرط التكرار صحيح) تنفذ العملية A ، ويؤخذ قرار التكرار مرة أخرى .



شكل (٧ - ٥)

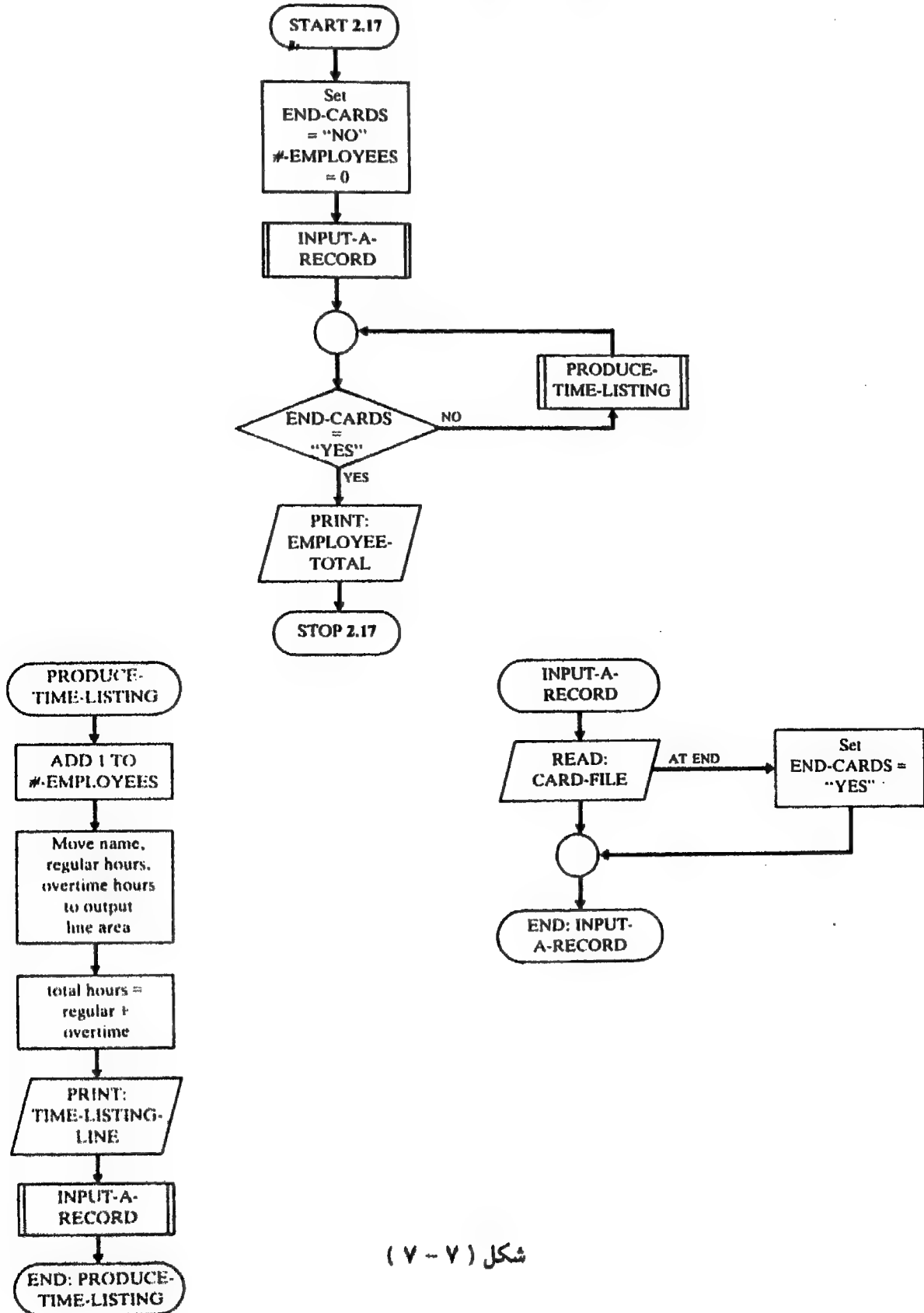
ويبين شكل ٧ - ٦ ميكالاً خاصاً DO WHILE ، يتسبب في وضع A مساوية NO ، وطباعة PRINT عبارة This really Prints وذلك مرة واحدة . الحقيقة إن A توضع فيها NO أثناء أول تنفيذ (وهو التنفيذ الوحيد) لا تتسبب في فصل هذا التنفيذ . وعندما يعاد الدخول في المجموعة فقط .. يتسبب الشرط "A = "NO" في تجنب عمل أى تكرار آخر .



شكل (٧ - ٦)

مثال ٧ - ٤ :

خريطة المسار المرتبة للبرنامج مثال رقم ١٧ من الفصل الثاني معطاة في شكل ٧ . ٧ ، وتشمل هياكل تتابع وتكرار فقط .



شكل (٧ - ٧)

لاحظ كيف توصف كل عملية سابقة التعريف في خريطة المسار الخاصة بها ، هذه العمليات تناظر مقاطع PERFORM في برنامج الكويل . لاحظ كذلك أن مقطعاً معيناً من جزء الإجراءات يعمل ، مثل فتح وإغلاق الملفات ، ليس مرسوماً في خريطة المسار .

٧ - ٣ هيكل التتابع في الكويل

هيكل التتابع هو الأسهل في البرمجة (بأى لغة) . يتحقق التتابع في الكويل بكتابة عبارة بعد الأخرى في جزء الإجراءات ببساطة .

٧ - ٤ هيكل الاختيار في الكويل

ينفذ هيكل الاختيار في الكويل بعبارة إذا IF ، والتي لها الشكل العام التالي :

$$\text{IF condition} \left\{ \begin{array}{l} \text{statement-1} \\ \text{NEXT SENTENCE} \end{array} \right\} \left[\begin{array}{l} \text{ELSE statement-2} \\ \text{ELSE NEXT SENTENCE} \end{array} \right]$$

هنا يكون الشرط Condition تعبيراً أو جزءاً من عبارة معقداً قليلاً أو كثيراً ، وبتناوله بالتفصيل في الأقسام من ٥ إلى ٩ في هذا الفصل.

مثال ٧ - ٥

```
IF HOURS-WORKED GREATER THAN 40.0
  COMPUTE GROSS-PAY = 40.0 * HOURLY-RATE +
    (HOURS-WORKED - 40.0) * 1.5 * HOURLY-RATE
ELSE
  COMPUTE GROSS-PAY = HOURS-WORKED * HOURLY-RATE
```

الشرط هو : HOURS - WORKED GREATER THAN 40.0 .

ويكون صحيحاً أو خطأ طبقاً للمحتويات الحالية في العنصر HOURS - WORKED . فإذا كان صحيحاً ، تنفذ state-ment - 1 وهي COMPUTE ... 40.0... ، ثم يتجه الكمبيوتر بعد ذلك لتنفيذ العبارة التي تلي النقطة التي تنهى عبارة IF . إذا كان الشرط خطأ ، تهمل Statement - 1 ، وتنفذ Statement - 2 وهي COMPUTE ... HOURS - WORKED ... ، ثم يتجه الكمبيوتر بعد ذلك إلى العبارة التي تلي النقطة التي تنهى عبارة IF .

بالرغم من عدم ضرورة ترجمة الترحيل الموضح ، إلا أنه هو طريقة نمطية ضرورية لكتابة الشفرة . كتبت IF و ELSE في نفس العمود ، وتم ترحيل جزء IF وجزء ELSE على التوالي تحت IF و ELSE . يجب استخدام نقطة مرتبة دائماً في تحديد نهاية كل عبارة من عبارات IF .

جزء ELSE في عبارة IF اختياري . فإذا حذف .. يتسبب الشرط الصحيح في تنفيذ Statement - 1 قبل أن ينتقل الكمبيوتر إلى العبارة التي تلي النقطة التي تنهى عبارة IF ، بينما يتسبب الشرط الخطأ في ترك الكمبيوتر Statement - 1 والانتقال مباشرة إلى العبارة التالية للنقطة التي تنهى عبارة IF .

مثال ٧ - ٦ : ليس ضروريا على الإطلاق تحديد ELSE NEXT SENTENCE ، حيث إن :

```
IF ...
  PERFORM ...
ELSE
  NEXT SENTENCE

WRITE ...
```

يتطابق مع

```
IF ...
  PERFORM ...

WRITE ...
```

مع أى من التشكيلين ، تنفذ PERFORM ثم WRITE إذا كان الشرط صحيحا ، بينما تنفذ WRITE فقط إذا كان الشرط خطأ .

إلا أن ELSE NEXT SENTENCE يمكن أن تكون مريحة في بعض الأحيان ، انظر مثال ٧ - ٢٥ .

مثال ٧ - ٧

```
IF AMOUNT-SOLD IS GREATER THAN SALES-QUOTA
  NEXT SENTENCE
ELSE
  MOVE "YES" TO SEND-MOTIVATING-LETTER-SWITCH
  SUBTRACT AMOUNT-SOLD FROM SALES-QUOTA GIVING AMOUNT-UNDER-QUOTA

WRITE REPORT-LINE
FROM WS-SALES-PERFORMANCE-LINE
```

إذا كانت قيمة AMOUNT - SOLD أكبر من SALES - QUOTA ، تنفذ NEXT SENTENCE ، متسببة في انتقال الكمبيوتر إلى العبارة التي تلي النقطة التي تنهى عبارة IF ، وهى WRITE PERIOD - LINE . إذا لم تكن AMOUNT - SOLD أكبر من SALES - QUOTA ، فيهمل الكمبيوتر جزء IF وينتقل إلى جزء ELSE منفذا MOVE و SUBTRACT قبل الانتقال إلى عبارة WRITE .

لاحظ ما يتبع حذف النقطة من نهاية عبارة IF : تصبح عبارة WRITE جزء من ELSE ، ويطبع سطر تقرير لكل بائع من البائعين الذين لم يحققوا حصص بيعهم فقط ، بدلا من طباعة سطر تقدير لكل بائع من البائعين .

مثال ٧ - ٨ :

ليس ضروريا - على الإطلاق أن يحدد NEXT SENTENCE في جزء IF من عبارة IF : فيمكن نفى الشرط واستبدال جزء IF مع جزء ELSE دائما . بعد عمل ذلك ، يحتوى جزء ELSE على NEXT SENTENCE والذي يمكن اهماله (انظر مثال ٦ - ٧) .

بتطبيق هذا المبدأ على مثال ٧ - ٧ نحصل على :

IF AMOUNT-SOLD IS NOT GREATER THAN SALES-QUOTA
MOVE "YES" TO SEND-MOTIVATING-LETTER-SWITCH
SUBTRACT AMOUNT-SOLD FROM SALES-QUOTA GIVING AMOUNT-UNDER-QUOTA

WRITE REPORT-LINE
FROM WS-SALES-PERFORMANCE-LINE

هناك خلل بسيط في عبارة IF - ومكافئها ، مثال ٧ - ٧ ، وهو انه ينتج خطاب تحريك ، يرسل عندما تساوى الكمية الأقل من الحصة صفراً . وتصحيح ذلك سهل :

IF AMOUNT-SOLD IS LESS THAN SALES-QUOTA

٧ - ٥ هيكل الاختيار : شرط الفئة

يمكن استخدام شرط الفئة class condition في تحديد ما - اذا كان عنصر البيانات يحتوى على بيانات حرفية أو عددية . وتكوين هذا الشرط هو ما يلي :

identifier IS [NOT] {
 NUMERIC
 ALPHABETIC
}

يمكن تطبيق اختبار NUMERIC واختبار NOT NUMERIC على عناصر بيانات تكون صورها X أو SV9 فقط ، ويكون استخدامها DISPLAY (أو 3 - COMP إذا كان 3 - COMP متاحاً) . يمكن تطبيق اختبار ALPHABETIC واختبار NOT ALPHABETIC على عناصر بيانات ، تكون صورها X أو A فقط واستخدامها DISPLAY .

الشرط identifier IS NUMERIC

يكون صحيحاً إذا كان ، وإذا فقط ، احتوى العنصر على أرقام من 0 إلى 9 . إذا شملت الصورة PIC اشارة عمليات (S) ، فيجب أن يحتوى العنصر كذلك على اشارة عمليات صحيحه . فإذا لم توجد اشارة فى الصورة ، فيجب ألا يحتوى العنصر نفسه على إشارة عمليات .

ويكون الشرط : identifier IS ALPHABETIC

صحيحاً إذا ، وإذا فقط ، احتوى العنصر على حروف من A إلى Z وفراغات فقط .

الاستخدام الاساسى لاختبار الفئة يكون للتأكد من صحة بيانات المدخلات فى السجلات التي يتم إدخالها بواسطة الشخص المُشغل . والبرنامج الذى يجرى الاختبار يجب أن ينتج رسائل خطأ وصفية بها معلومات كافية للسماح بتصحيح الخطأ.

مثال ٧ - ٩

```
IF PARTIAL-PAYMENT-AMOUNT NUMERIC
  SUBTRACT PARTIAL-PAYMENT-AMOUNT FROM OLD-BALANCE GIVING
    NEW-BALANCE
ELSE
  MOVE OLD-BALANCE TO NEW-BALANCE
  MOVE "YES" TO UNPROCESSED-PAYMENT-SWITCH
```

حقل المدخلات PARTIAL - PAYMENT - AMOUNT يكتبه شخص مشغل ، ويتم التأكد من صحته باختبار فئة قبل استخدامه في الحسابات. ولم يتم التأكد من صحة OLD - BALANCE ، لأنه يأتي من ملف قرص يتم إنتاجه ببرنامج كمبيوتر . فإذا كان البرنامج مصمماً بطريقة مناسبة وخالياً من الأخطاء ، فيفترض في الملفات التي ينتجها البرنامج أنها تحتوى بيانات صحيحة (حيث يفترض أن البرنامج اختبر صحة كل البيانات التي وضعت في الملف) .

٧- ٦ هيكل الاختيار : شرط علاقة

إن تكوين الشرط العلاقي relation condition هي كما يلي :

$$\text{operand-1 IS [NOT] } \left\{ \begin{array}{l} \text{GREATER THAN} \\ > \\ \text{LESS THAN} \\ < \\ \text{EQUAL TO} \\ = \end{array} \right\} \text{operand-2}$$

باستثناء بعض الخليط الممنوع (انظر مثال ٧ - ١٢) يمكن أن يكون كل من العامل الأول operand - 1 و العامل الثاني operand - 2 عنصر بيانات ، أو ثابتاً ، أو تعبيراً حسابياً (مثل المسموح به في الطرف الأيمن لعبارة COMPUTE) .

مثال ٧ - ١٠

- IF A + B ** 2 LESS THAN (A - C) * (A + B)

أى تعبير حسابي صحيح في عبارة COMPUTE يكون صحيحاً في شرط العلاقة كذلك .

- IF EMPLOYEE-ID-NUMBER EQUAL HIGH-VALUES

عادة ما تستخدم الثوابت الاستعارية في العلاقات .

- IF GROSS-PAY GREATER THAN 900.00

استخدام ثابت عددي في شرط علاقة .

- IF INPUT-RECORD-CODE = "A"

في الكويل .. يجب التعبير عن العلاقات « أقل من أو يساوي » و « أكبر أو يساوي » بأنها نفي العلاقة العكسية .

- IF INPUT-RECORD-CODE NOT EQUAL "C"

IS و TO هي كلمات اختيارية في شرط العلاقة .

في الكويل ، ماذا يعني القول إن أحد العناصر أقل من عنصر آخر ؟

مقارنات عددية : لا يؤثر الاستخدام USAGE ، وحجم الصورة PICTURE على نتائج المقارنة ، والتي تكون مبنية تماما على القيم الجبرية للعناصر . تضبط العلامات العشرية تلقائيا لأغراض المقارنة . (نفس القواعد التي تحسن من كفاءة الحسابات تحسن كذلك من كفاءة المقارنات : أنظر القسم 18 من الفصل السادس) .

مثال ٧ - ١١ معرفة ما يلي :

01 SAMPLE-DATA-ITEMS.

05	A	PIC S9999V99	COMP-3	VALUE +25.67.
05	B	PIC S99V999	COMP	VALUE +25.670.
05	C	PIC 99V99	DISPLAY	VALUE 25.67.
05	D	PIC S99V99		VALUE -99.00.

تعتبر كل من A و B و C متساوية (فهي تحتوى جميعها على نفس القيمة الجبرية 25.6 +) D أقل من أى من A, B, C.

مقارنات غير عددية : هنا تعرف « أقل من » طبقا لتسلسل التتابع أثناء تنفيذ البرنامج (انظر القسم 3 من الفصل الرابع والقسم 4 من الفصل الرابع كذلك) . تجرى المقارنة غير العددية على بايت واحد في نفس الوقت ، مع العمل من اليسار إلى اليمين للعنصرين المقارنين . فإذا ما اتفق العنصران على الطريق فيعتبران متساويين . وإذا حدث عدم اتفاق في بعض المواقع ، فتحدد النتيجة طبقا للترتيب النسبي للبايت غير المتوافق في تسلسل التتابع . إذا اختلف طول العنصرين المقارنين ، فيعتبر الأقصر معتدا بفراغات من ناحية اليمين .

مثال ٧ - ١٢

(i) CAB أقل من CAT لأنه بالرغم من توافق أول 2 بايت ، إلا أن البايت الثالث مختلف ، B أقل من (تسبق) T في تسلسل التتابع .

(ب) CAB تساوى CABbb ، حيث إن CAB (عنصر أقصر) يعتبر معتدا بفراغين من ناحية اليمين لغرض المقارنة.

(ج) COB أكبر من CAT لحدوث عدم توافق في الموقع الثانى ، O أكبر من A في تسلسل المقارنة . لاحظ أن B و T لن تقارنا على الإطلاق ، حيث إن النتيجة تحددت بالبايت الثانى .

(د) CAB أقل من CABBb لان CAB يعتبر ممتدا بقراغين من ناحيه اليمين لأغراض المقارنة . ويحدث أول عدم توافق في أول فراغ من هذين القراغين (CABBb تقارن مع CABBb). وحيث إن الفراغ أقل من B في تسلسل المقارنة ، فيكون CAB أقل من CABBb .

لقد رأينا أن المقارنات غير العددية تحتفظ بالترتيب المعتاد لحروف الهجاء .

مثال ٧ - ١٣ :

كما هو الحال مع عبارة MOVE ، من الممنوع خلط بعض العوامل غير العددية والعددية في شرط العلاقة . لاحظ ما يلي ، بصفة خاصة :

(١) يمكن مقارنة العناصر العددية المعرفة باستخدام COMP و COMP - 3 ، مع عناصر عددية فقط (والتي يمكن أن تعرف بأنها DISPLAY أو COMP و COMP - 3) .

(٢) لا يمكن مقارنة الثوابت أو الثوابت العددية مع بعضها البعض .

(٣) يمكن مقارنة العناصر العددية المعرفة بأنها DISPLAY مع عناصر حرفية عددية PIC X - إلا أن النتائج تختلف عما هو متوقع لها . إذا فكرت في أداء مثل هذه المقارنة ، فعليك بفحص دليل الكويل المحدد المتاح لك .

٧ - ٧ هيكل الاختيار : شرط الإشارة

يمكن استخدام شرط الإشارة sign condition مع عوامل عددية فقط ، وتكوينه كما يلي :

operand IS [NOT] { POSITIVE
ZERO
NEGATIVE }

مثال ٧ - ١٤

- IF CURRENT-BALANCE IS POSITIVE

الشرط صحيح إذا كانت الموازنة أكبر من صفر فقط . وفيما يلي عبارة مكافئة لهذه العبارة تماما

IF CURRENT-BALANCE IS GREATER THAN ZERO

- IF CURRENT-BALANCE - PAYMENT-AMOUNT IS NOT NEGATIVE

استخدم تعبيراً حسابياً في شرط إشارة ، وهو مكافئ لما يلي :

IF CURRENT-BALANCE - PAYMENT-AMOUNT IS NOT LESS THAN ZERO

- IF NUMBER-OVERDUE-ACCOUNTS IS ZERO


```

IF WS-END-OF-FILE-SW NOT EQUAL "YES"
  IF CUSTOMER-UPDATE-CODE EQUAL "A"
    PERFORM ADD-NEW-CUSTOMER-ROUTINE
  ELSE IF CUSTOMER-UPDATE-CODE EQUAL "D"
    PERFORM DELETE-A-CUSTOMER-ROUTINE
  ELSE IF CUSTOMER-UPDATE-CODE EQUAL "C"
    PERFORM CHANGE-NAME-ADDRESS-ROUTINE
  ELSE
    PERFORM INVALID-UPDATE-CODE-ROUTINE

```

يحتوى أول بايت من CUSTOMER - UPDATE - RECORD على رمز يحدد ما إذا كان السجل المجدد محتويا على معلومات عن عميل جديد ليضاف إلى CUSTOMER - MASTER - FILE ، أو معلومات عن عميل يحذف من الملف ، أو معلومات عن عميل يجب تغيير اسمه وعنوانه في الملف . ويختبر هذا الرمز بسلسلة من عبارات IF ، التي تحدد المقطع المناسب لتنفيذه . بهدف تشغيل سجل العمليات الجارية التالي . وهناك الملاحظتان التاليتان :

(١) تظهر قيم الرمز ككوابيت في عبارات IF ، فإذا كان مطلوبا تغيير هذه القيم ، فيجب أن يبحث المبرمج خلال جزء الاجراءات لإيجاد مواقع حدوث الثوابت.

(٢) لا يحمل "C" CUSTOMER - UPDATE - CODE EQUAL "C" معلومات كافية عما يعنيه عندما يكون الرمز مساويا 'C'

(ب)

```

01 CUSTOMER-UPDATE-RECORD.
05 CUSTOMER-UPDATE-CODE          PIC X.
   88 ADD-NEW-CUSTOMER-TRANSACTION VALUE "A".
   88 DELETE-CUSTOMER-TRANSACTION  VALUE "D".
   88 CHANGE-ADDRESS-TRANSACTION   VALUE "C".
05 UPDATE-CUSTOMER-NAME          PIC X(20).
05 UPDATE-CUSTOMER-ADDRESS       PIC X(30).
.....
01 WS-END-OF-FILE-SW             PIC X(3).
   88 NO-MORE-TRANSACTION-RECORDS  VALUE "YES".
   88 TRANSACTION-RECORD-WAS-INPUT VALUE "NO".
.....
READ CUSTOMER-UPDATE-FILE
AT END
  MOVE "YES" TO WS-END-OF-FILE-SW

IF TRANSACTION-RECORD-WAS-INPUT
  IF ADD-NEW-CUSTOMER-TRANSACTION
    PERFORM ADD-NEW-CUSTOMER-ROUTINE
  ELSE IF DELETE-CUSTOMER-TRANSACTION
    PERFORM DELETE-A-CUSTOMER-ROUTINE
  ELSE IF CHANGE-ADDRESS-TRANSACTION
    PERFORM CHANGE-NAME-ADDRESS-ROUTINE
  ELSE
    PERFORM INVALID-UPDATE-CODE-ROUTINE

```

جزء VALUE مطلوب لعناصر المستوى 88 ، وغير مسموح بأى أجزاء أخرى . هذا له معنى ؛ لأن الاسم الشرطى تصاحبه قيمه خاصة لعنصر البيانات المعرف الاسم الشرطى تحته .

لاحظ كيفية استبدال شروط العلاقة من (أ) بشروط أسماء شرطية . مثل استبدال :

IF CUSTOMER-UPDATE-CODE EQUAL "A"

بمكافئها الكامل التالى :

IF ADD-NEW-CUSTOMER-TRANSACTION

تتحقق الثلاث معيزات الرئيسية من استخدام شروط الأسماء الشرطية التالية :

- (١) بدون استخدام ثوابت فى عبارات IF و PERFORM من جزء الاجراءات .. يكون تعديل البرنامج أسهل إذا كان معنى رموز المدخلات أو الرموز نفسها يتغير . وكل ما يتطلب تعديل هو محتويات المستوى 88 فى جزء البيانات .
- (٢) يمكن أن تعد الأسماء الشرطية بحيث تكون وصفيه للشرط الذى تصفه . وعلى هذا يحمل - ADDRESS - CHANGE TRANSALTION معنى أكبر للقارئ عن "C" CUSTOMER - UPDATE - CODE EQUAL .
- (٣) يمكن للاسم الشرطى أن يمثل ببساطة شرط علاقة أكثر تعقيدا .

مثال ٧ - ١٦

01	TRANSCRIPT-RECORD.	
05	TRANSCRIPT-STUDENT-ID	PIC X(9).
05	TRANSCRIPT-YEAR-ATTENDING	PIC 9.
88	FRESHMAN	VALUE 1.
88	SOPHOMORE	VALUE 2.
88	JUNIOR	VALUE 3.
88	SENIOR	VALUE 4.
88	UNDERGRADUATE	VALUE 1 THRU 4.
88	GRADUATE	VALUE 5 THRU 9.
05	TRANSCRIPT-CREDITS-COMPLETED	PIC 99.
05	TRANSCRIPT-GRADE-POINT-AVERAGE	PIC 9V9.
88	HIGHEST-DISTINCTION	VALUE 3.8 THRU 4.0.
88	HIGH-DISTINCTION	VALUE 3.5 THRU 3.7.
88	NOT-GRADUATING	VALUE 0.0 THRU 1.9.
05	TRANSCRIPT-STUDENT-NAME	PIC X(30).
.....		

يسمح استخدام جزء THRU بتحديد مدى range قيم للاسم الشرطى . يكون الشرط UNDER - GRADUATE

صحيحا إذا كانت قيمة TRANSCRIPT - YEAR - ATTENDING تقع بين 1 و 4 بما فى ذلك الحدود ... إلخ .

يفضل بعض المبرمجين كتابة المستوى 88 فى المنطقة A (كما سبق عمل ذلك فى المثال السابق) ، لجعله موجودا بوضوح

، ويفضل البعض الآخر ترحيله كالعتاد (كما فى مثال ٧ . ١٥ - الجزء ب) .

لاحظ الآن بساطة وسهولة فهم ما يلي :

IF SENIOR AND HIGH-DISTINCTION

عن شرط العلاقة .. المناظر له التالي

IF TRANSCRIPT-YEAR-ATTENDING EQUAL 4 AND
TRANSCRIPT-GRADE-POINT-AVERAGE NOT LESS THAN 3.5 AND
TRANSCRIPT-GRADE-POINT-AVERAGE NOT GREATER THAN 3.7

مثال ٧ - ١٧ : يمكن تحديد أكثر من مدى قيم واحد لاسم شرطي معين .

05 LOCATION-CODE PIC XX.
88 PHILADELPHIA VALUE "AA" THRU "AC"
"B7" "D5"
"E3" THRU "E5"

بمعرفه التعريف السابق .. يمكن استخدام ما يلي :

IF PHILADELPHIA

بدلا من التعقيد التالي

IF (LOCATION-CODE NOT LESS THAN "AA"
AND NOT GREATER THAN "AC")
OR (LOCATION-CODE EQUAL "B7" OR "D5")
OR (LOCATION-CODE NOT LESS THAN "E3"
AND NOT GREATER THAN "E5")

مثال ٧ - ١٨ : بمعرفه التعريفات التالية :

01 END-OF-FILE-SWITCH PIC XXX.
88 NO-MORE-RECORDS VALUE "YES".
88 MORE-RECORDS VALUE "NO".

فإننا نلاحظ خطأين يقع فيهما المبتدئون :

• READ SAMPLE-FILE
AT END
MOVE "YES" TO NO-MORE-RECORDS

(في موقع MOVE "YES" TO END - OF - FILE - SWITCH) . الاسم الشرطي ليس هو عنصر البيانات ، إلا أنه تمثيل شرط علاقة يشمل عنصر بيانات . والعباره التالية :

MOVE "YES" TO NO-MORE-RECORDS

تعطى نفس الإحساس تماما مثل العبارة التالية :

MOVE "YES" TO (END-OF-FILE-SWITCH IS EQUAL TO "YES")

- IF NO-MORE-RECORDS IS EQUAL TO "YES"

يكون NO - MORE - RECORDS شرط علاقة فعلا (فى صورة موجزة) ، وما يؤديه هو :

IF NO-MORE-RECORDS

أو

IF END-OF-FILE-SWITCH IS EQUAL TO "YES"

٧ - ٩ المؤثرات المنطقية والشروط المركبة

يمكن استخدام المؤثر المنطقي الأحادي NOT فى نفي شرط بسيط simple condition (شرط فئة أو علاقة أو إشارة أو اسم شرطى) ، بافتراض أن الشرط البسيط نفسه لا يحتوى على NOT .

مثال ٧ - ١٩

- IF NOT A IS LESS THAN B

الشرط المنفى هو A IS LESS THAN B ، وعلى هذا .. يمكن أن يكون ما يلى تعويضا أوضح لعبارة IF :

IF NOT (A IS LESS THAN B)

أى صيغة مكافئة منطقيا لما يلى :

IF A IS NOT LESS THAN B

تفضل هذه الصيغة الأخيرة ، الموجودة بها NOT فى شرط بسيط ، بالنسبة إلى شروط الفئة والاشارة والعلاقة .

- IF NOT RECORD-IN-BUFFER

"RECORD-IN-BUFFER" must be a condition-name. If RECORD-IN-BUFFER is true, NOT RECORD-IN-BUFFER is false; if RECORD-IN-BUFFER is false, NOT RECORD-IN-BUFFER is true.

بالإضافة إلى النفى يمكن دمج الشروط البسيطة ، باستخدام المؤثرات العلاقية OR و AND ؛ لتكوين شروط مركبة com-

conditions 1 AND conditions 2 " لـ القيمة الحقيقية لـ conditions 1 AND conditions 2 " تعتمد على القيم الحقيقية لـ conditions 1 AND conditions 2 كما هو موضح في شكل ٧ - ٨ .

condition-1 AND condition-2	condition-2	condition-1
صحيح	صحيح	صحيح
خطأ	خطأ	صحيح
خطأ	صحيح	خطأ
خطأ	خطأ	خطأ

شكل (٧ - ٨)

لاحظ إنه إذا ما ارتبط شرطان بالمؤثر AND .. فيكون الشرط المركب الناتج عن ذلك صحيحا ، عندما يكون كل من الشرطين البسيطين الأصليين صحيحا فقط .
عندما يرتبط شرطان مع بعضهما باستخدام المؤثر OR ، فيكون الشرط المركب جدول الحقيقة الموجود في شكل ٧ - ٩ .

condition - 1 OR condition - 2	condition - 2	condition - 1
صحيح	صحيح	صحيح
صحيح	خطأ	صحيح
صحيح	صحيح	خطأ
خطأ	خطأ	خطأ

شكل (٧ - ٩)

لاحظ أن الشرط المركب يكون خطأ إذا كان كل من الشرطين البسيطين خطأ فقط .

مثال ٧ - ٢٠

- IF HOURS-WORKED NUMERIC AND HOURS-WORKED GREATER THAN 40.0

الشرط المركب يكون صحيحا إذا كان ، وإذا كان فقط ، الشرطان البسيطان صحيحين . يمكن أن يحدث هذا الشرط الخاص مشاكل، ويتم تقويم شروط العلاقات في كويل IBM OS/VS قبل شروط الفئات . وعلى هذا .. كان HOURS WORKED غير عددي ، يمكن أن يتسبب تقويم HOURS - WORKED GREATER THAN 40.0 في إنهاء البرنامج نهاية غير عادية . ويمكن حذف هذه الصعوبة باستخدام عبارات IF :

```
IF HOURS-WORKED NUMERIC
  IF HOURS-WORKED GREATER THAN 40.0
```

الآن .. تم تقويم شرط العلاقة عندما كان HOURS - WORKED عدديا فقط .

- IF HOURS-WORKED NOT NUMERIC OR HOURLY-RATE NOT NUMERIC
PERFORM ERROR-ROUTINE
ELSE
PERFORM CALCULATE-PAY

إذا كان أحد الشروط البسيطة صحيحا على الأقل .. فإن الشرط المركب يكون صحيحا . وعلى هذا .. يتخذ المقطع ER - ROUTINE إذا كان أى من العنصرين أو كلاهما ، غير عددي HOURS WORKED و HOURLY - RATE ، أما إذا كان العنصران عدديين فينفذ CALCULATE - PAY .

- IF HOURS-WORKED NUMERIC AND HOURLY-RATE NUMERIC
PERFORM CALCULATE-PAY
ELSE
PERFORM ERROR-ROUTINE
- وهو مكافئ تماما لما سبق . يجب أن تتحقق (بواسطة جداول الحقيقة) أن العكس (أى النقي) لـ A OR B هو (NOT A) AND (NOT B) ، وأن عكس A AND B هو (NOT A) OR (NOT B) .

- IF RECORD-WAS-INPUT AND BALANCE DUE IS POSITIVE

شرط الاسم الشرطي هنا ، وشرط الإشارة مرتبطان بالمؤثر AND .

- IF ON-HAND NOT LESS THAN 1 OR ON-HAND NOT GREATER THAN 500
PERFORM ADJUST-QUANTITY-ONHAND

كتابه خاطئة . الهدف كان (كما نتوقع) تنفيذ PERFORM ADJUST - QUANTITY - ON - HAND ، إذا لم تقع محتويات ON - HAND بين 1 أو 500 مع شمول الحدين . إلا أنه بشيء من التأمل .. نرى أن الشرط المركب صحيح دائما (الطريقة الوحيدة التي يمكن أن يكون بها خطأ ، هي أن يكون ON - HAND أقل من 1 وأكبر من 500 في نفس الوقت - وهذا أمر مستحيل) ، وعلى هذا .. تنفذ PERFORM دائما . الشفرة الصحيحة هي :

```
IF ON-HAND LESS THAN 1 OR ON-HAND GREATER THAN 500
  PERFORM ADJUST-QUANTITY-ONHAND
```

يمكن استخدام أكثر من مؤثر منطقي واحد لعمل شرط مركب . عندما يتكرر نفس المؤثر المنطقي... تكون الأقواس ضرورية في الشرط المركب (حيث تكون النتائج هي نفسها دائما ، بغض النظر عن ترتيب تطبيقات المؤثر الفردي) . وعندما تستخدم مؤثرات منطقية مختلفة .. فيجب أن يكون ترتيب تقويمها محددًا بوضوح بواسطة الأقواس .

مثال ٧ - ٢١

- IF INPUT-YEAR IS GREATER THAN 80
AND INPUT-YEAR IS LESS THAN 84
AND INPUT-MONTH IS NOT LESS THAN 1
AND INPUT-MONTH IS NOT GREATER THAN 12
AND INPUT-DAY IS NOT LESS THAN 1
AND INPUT-DAY IS NOT GREATER THAN 31
PERFORM PROCESS-VALID-DATE
ELSE
PERFORM PROCESS-INVALID-DATE

لاحظ أن كل شرط بسيط مكتوب في سطر خاص به ، وأن الشروط والمؤثرات مضبوطة لتسهيل القراءة . كل الستة شروط البسيطة تكون صحيحة ؛ لكي يكون الشرط المركب صحيحا . وعلى هذا .. لكي تنفذ - PERFORM PROCESS - VALID - DATE ، يجب أن تقع السنة بين 83,81 مع شمول الحدين ، ويجب أن يقع الشهر بين 1 و 12 ، ويجب أن يقع اليوم بين 1 و 31 . بينما يكتشف هذا الاختبار معظم الأخطاء في حقل التاريخ ، فهو يترك بعض الأخطاء تمر (مثل 09 / 31 / 83)

- IF TIME-CARD-EMPLOYEE-NUMBER NOT NUMERIC
OR TIME-CARD-REGULAR-HOURS NOT NUMERIC
OR TIME-CARD-OVERTIME-HOURS NOT NUMERIC
MOVE "YES" TO INVALID-NUMERIC-FIELDS-SW

يتأكد هذا المقطع من صحة حقول عديدة في TIME - CARD - RECORD لاحظ كيف يكتب كل شرط بسيط على سطر خاص به ، مع ضبط المؤثرات لتسهيل القراءة . وحيث إن OR هو المستخدم .. فإن الشرط المركب يكون صحيحا إذا كان أحد الشروط البسيطة أو أكثر صحيحا .

- IF (NOT SPECIAL-TERMS-REQUESTED)
AND (CREDIT-UNSATISFACTORY
OR PURCHASE-AMOUNT GREATER THAN CREDIT-LIMIT)

يجب أن يكون CREDIT - UNSATISFACTORY , SPECIAL - TERMS - REQUESTED أسماء شرطية . ترتيب التقويم ، هو : (١) التقويم SPECIAL - TERMS - REQUESTED . (٢) تنفي نتيجة (١) . (٣) تقويم CREDIT - PURCHASE - AMOUNT GREATER THAN CREDIT - LIMIT (٤) تقويم CREDIT - UNSATISFACTORY . (٥) تربط نتيجة (٣) ونتيجة (٤) بواسطة OR . (٦) تربط نتيجة (٢) ونتيجة (٥) بواسطة AND .

- IF (NEW-CUSTOMER
AND (CUSTOMER-ID-ON-FILE OR
CUSTOMER-ID NOT NUMERIC))
OR
(OLD-CUSTOMER
AND (CUSTOMER-ID-NOT-ON-FILE
OR DAYS-PAYMENT-LATE GREATER THAN 90))

إذا أصبحت الشروط المركبة معقدة بهذه الدرجة ، فهذه اشارة طيبة أنه يجب تجزئة البرنامج ، أو يجب إعادة تصميمه .
إذا لم تستخدم الأقواس (أو إذا كانت الشروط فى نفس مستوى الأقواس) ، فيتم تقويم التعبيرات المنطقية بالترتيب
التالى : (١) التعبيرات الحسابية التى تظهر فى شروط العلاقات . (٢) الشروط البسيطة بالترتيب : العلاقة فالفئة فالاسم
الشروطى فالإشاره (٣) الشروط البسيطة المنفية بنفس الترتيب السابق ذكره . (٤) AND (٥) OR . إذا كان هناك شرطان
جزئيان متساويان طبقاً لترتيبهما ، فيتم تقويمهما من اليسار لليمن .
إذا بدت القواعد السابقة معقدة ، فهناك طريقة بسيطة لحذف أبعد امكانية لخطأ بسبب التقويم غير الصحيح لتعبير
منطقى مركب : استخدم الأقواس .

مثال ٧ - ٢٢ :

بدون أقواس ، يقرأ ثالث شرط مركب فى مثال ٧ - ٢١ على النحو التالى :

IF NOT SPECIAL-TERMS-REQUESTED
AND CREDIT-UNSATISFACTORY
OR PURCHASE-AMOUNT GREATER THAN CREDIT-LIMIT

ترتيب التقويم هو كما يلى : (١) شرط العلاقة PURCHASE - AMOUNT GREATER THAN CREDIT - LIM-

. IT

(٢) شرط اسم شرطى ، من اليسار لليمن SPECIAL - TERMS - REQUESTED .

(٣) شرط اسم شرطى ، من اليسار لليمن CREDIT - UNSATISFACTORY .

(٤) نفى شرط بسيط NOT لرقم (٢) .

(٥) ربط (٢) و (٤) بالمؤثر AND (AND تنفذ قبل OR)

(٦) ربط (١) و (٥) بواسطة OR .

والنتيجة هى كما لو استخدمت الأقواس على النحو التالى :

IF ((NOT SPECIAL-TERMS-REQUESTED)
AND CREDIT-UNSATISFACTORY)
OR (PURCHASE-AMOUNT GREATER THAN CREDIT-LIMIT)

وعلى هذا .. تكون الأقواس في المثال ٧ - ٢١ ضرورية لتحقيق النتيجة المرجوة (والتي تختلف عن النتيجة المذكورة عالياً).

٧ - ١٠ اختصار شروط علاقات مركبة

شروط العلاقة البسيط العام له الشكل التالي : subject relation - operator object

يدمج شرط العلاقة المركب شرطى علاقات مركبين أو أكثر بواسطة المؤثرات العلائقية AND و OR .

subject relation - operator object { AND
OR }

subject relation - operator object

إذا كان المؤثر العلائقي relational - operator هو نفسه لشرطى علاقات بسيطتين في شرط علاقة مركب ، فلا تكون هناك حاجة لتكرار المؤثر العلائقي . وبالمثل .. إذا كان subject هو نفسه في عديد من العلاقات البسيطة لعلاقة مركبة ، فلا يكون هناك حاجة إلى تكراره . إلا أن object يجب أن تكتب دائماً لكل شرط علاقة بسيط .

مثال ٧ - ٢٣ :

- IF A EQUAL B OR A EQUAL C OR A EQUAL D

يتكرر فيها كل من subject (عنصر البيانات A) والمؤثر العلائقي (EQUAL) . ويمكن حذف التكرار لتجصل على ما يلي:

IF A EQUAL B OR C OR D

بالطبع .. يجب أن يجدد أول ظهور لـ subject والمؤثر العلائقي .

- IF A EQUAL B AND A EQUAL C OR A GREATER THAN D

يمكن اختصارها لتأخذ الشكل التالي :

IF A EQUAL B AND C OR GREATER THAN D

إذا لم يوجد subject أو المؤثر العلائقي في شرط العلاقة البسيط ، فيكون آخر subject أو آخر مؤثر علائقي يظهر في الشرط المركب مفهوماً .

- IF SAMPLE-ITEM NOT EQUAL 10.5 OR OTHER-ITEM OR 3.0

فى اختصار لما يلى :

```
IF SAMPLE-ITEM NOT EQUAL 10.5
  OR SAMPLE-ITEM NOT EQUAL OTHER-ITEM
  OR SAMPLE-ITEM NOT EQUAL 3.0
```

يجب أن يستخدم الشرط العلاقى المركب المختصر عندما لا يغيب الوضوح فقط ، بل عندما يكون هناك شك ، وعندئذ استخدم الاقواس ولا تختصر الشروط .

V - ١١ عبارة اذا المباشرة والمتداخلة

الخوارزمى المرتب المعد عن طريق دمج هياكل تتابع واختيار وتكرار عادة ما يحتوى على هياكل اختيار داخل هياكل اختيار . ونقول إن تنفيذ الكويل الذى له مثل هذا الخوارزمى بانه يحتوى على عبارات إذا متداخلة nested IF . عندما تستخدم إذا المتداخلة يجب أن يعكس الترحيل العلاقات الموجودة بين عبارات IF بدقة.

مثال ٧ - ٢٤ :

```
IF ADD-NEW-MASTER-RECORD
  IF MASTER-ALREADY-EXISTS
    PERFORM DUPLICATE-RECORD-ERROR
  ELSE
    PERFORM CREATE-NEW-MASTER-RECORD
  ELSE
    IF CHANGE-EXISTING-MASTER-RECORD
      IF MASTER-ALREADY-EXISTS
        PERFORM MAKE-CHANGES-TO-MASTER
      ELSE
        PERFORM RECORD-NOT-FOUND-ERROR
```

لاحظ كيف تم ترحيل كل عبارة IF ، وكيف ضببطت أجزاء ELSE مع IF المناظرة لها دائما ، لا توجد بصفة عامة أى مشكلة فى تفسير IF متداخلة ، عندما يصاحب كل IF جزء ELSE إلا أن هذا ليس لازما؛ عبارة IF CHANGE - EXIST - MASTER - RECORD ليس لها جزء ELSE .

مثال ٧ - ٢٥

عند تفسير عبارة IF متداخلة ، يرفق مترجم الكويل كل جزء ELSE مع أول IF تسبقه ، ولم يحدد لها جزء ELSE فعلا (إذا لم تتواجد IF غير مزدوجة مع ELSE فيحدث خطأ تكويني) . ويمكن أن تتسبب هذه القاعدة فى حدوث مشاكل ، عندما تكتب عبارات IF المتداخلة مع حذف أجزاء ELSE :

```
IF TRANSACTION-WAS-READ
  IF PAYMENT-AMOUNT NUMERIC
    SUBTRACT PAYMENT-AMOUNT FROM CURRENT-BALANCE
  ELSE
    MOVE "FINAL" TO TOTAL-LINE-HEADING
    PERFORM PRODUCE-FINAL-TOTAL-LINE
```

هنا .. يرفق المترجم ELSE مع IF PAYMENT - AMOUNT (بدلا من ارفاقها مع IF TRANSACTION - WAS READ - ، كما تبدو للمبرمج انه لديه ما يريد) . وعلى هذا .. تنفذ العبارات كما يلي :

```
IF TRANSACTION-WAS-READ
  IF PAYMENT-AMOUNT NUMERIC
    SUBTRACT PAYMENT-AMOUNT FROM CURRENT-BALANCE
  ELSE
    MOVE "FINAL" TO TOTAL-LINE-HEADING
    PERFORM PRODUCE-FINAL-TOTAL-LINE
```

عبارة IF المتداخلة السابقة غير صحيحة ، حيث إن سطر الإجمالي النهائي ينتج في أى وقت ، يتم إدخال دفعة غير عددية كجزء من سجل العمليات الجارية . كان الهدف الأساسى طباعة سطر الإجمالي النهائي عندما لا يمكن أن تتواجد سجلات عمليات جارية أخرى (أى نهاية الملف) .
لنم مثل هذه الاخطاء .. تتطلب بعض العمليات الجارية كتابة كل عبارة IF متداخلة بصاحبها جزء ELSE (حتى إذا لم تكن إلا ELSE NEXT SENTENCE) :

```
IF TRANSACTION-WAS-READ
  IF PAYMENT-AMOUNT NUMERIC
    SUBTRACT PAYMENT-AMOUNT FROM CURRENT-BALANCE
  ELSE
    NEXT SENTENCE
ELSE
  MOVE "FINAL" TO TOTAL-LINE-HEADING
  PERFORM PRODUCE-FINAL-TOTAL-LINE
```

إذا نفذت ELSE NEXT SENTENCE .. فانها تأخذ الكمبيوتر إلى العبارة التى تلى النقطة .
أجيانا .. يجب أن يكرر البرنامج اختبار نفس الحقل ليحدد أى من سلسلة قيم يحتويها . ويمكن استخدام صيغة خاصة من عبارة IF المتداخلة . تسمى IF الخطية linear فى مثل هذه الحالة .

مثال ٧ - ٢٦ :

```
IF TRANSACTION-CODE EQUAL 1
  PERFORM CREATE-NEW-MASTER-RECORD
ELSE
  IF TRANSACTION-CODE EQUAL 2
    PERFORM DELETE-MASTER-RECORD
  ELSE
    IF TRANSACTION-CODE EQUAL 3
      PERFORM CHANGE-EXISTING-MASTER-RECORD
    ELSE
      PERFORM INVALID-TRANSACTION-CODE-ROUTINE
```


يفضل كتابة سلسلة اختبارات TRANSACTION - CODE على صورة IF خطية كما يلي :

```
IF TRANSACTION-CODE EQUAL 1
  PERFORM CREATE-NEW-MASTER-RECORD
ELSE IF TRANSACTION-CODE EQUAL 2
  PERFORM DELETE-MASTER-RECORD
ELSE IF TRANSACTION-CODE EQUAL 3
  PERFORM CHANGE-EXISTING-MASTER-RECORD
ELSE
  PERFORM INVALID-TRANSACTION-CODE-ROUTINE
```

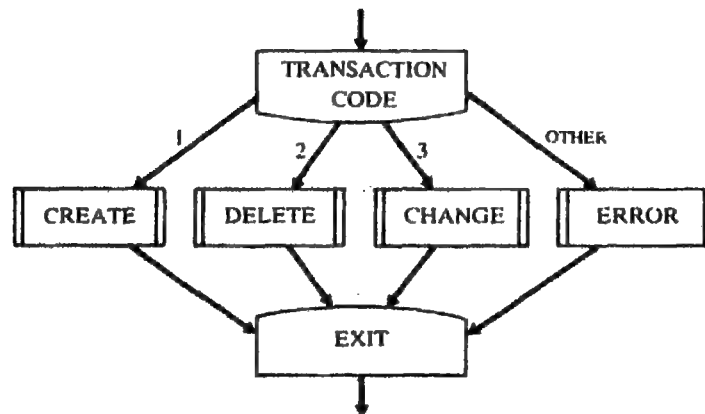
فى كل IF خطية .. تكتب كل ELSE تالية مضبوطة تحت IF الأصلي وتكتب كل IF تالية على نفس السطر الخاص بـ ELSE السابقة ، وترحل الإجراءات كالعادة . لاحظ أن كل جزء ELSE لا تتبعه IF . العبارة (العبارات) المصاحبة لآخر ELSE تنفذ إذا كان عنصر البيانات الذى يجرى عليه الاختبار لا يحتوى على أى قيمة من القيم المختبرة فى شروط العلاقة السابقة فقط . لاحظ كذلك انه ينفذ مقطعاً واحداً ، ومقطعاً واحداً فقط ، الذى يعتمد على محتويات TRANSACTION - CODE فقط .

مثال ٧ - ٢٧ :

يمكن استخدام IF الخطية فى كتابة هيكل رابع لمنطقة البرنامج وهو هيكل الحالة (مكون الحالة state structur) . وهيكل الحالة ليس ضرورياً على الإطلاق ، إلا أنه متاح فى بعض اللغات كوسيلة مريحة لعمل البرنامج . يتسبب هيكل الحالة فى تنفيذ أحد المقاطع بالضبط طبقاً لقيمة عنصر بيانات التحكم الذى يختبر داخل الهيكل . ويمكن رسم خريطة مسار لهيكل الحالة بعدة طرق ، التى يوضح شكل (٧ - ٧) إحداها ، وهى خريطة مسار للمثال رقم ٧ - ٢٦ .

٧ - ١٢ هيكل التكرار : عبارة التنفيذ

تسمح عبارة التنفيذ PERFORM بتنفيذ مقطع أو قسم من جزء الإجراءات ، صفر من المرات أو مرة واحدة ، أو أى عدد من المرات . ولها أربعة أشكال .



شكل (٧ - ١٠)

تنفيذ بسيط

أبسط صيغة لعبارة PERFORM هي :

PERFORM procedure-name-1 { **THROUGH** } procedure-name-2
THRU

كل اسم إجراءات procedure - name يجب أن يكون اسم مقطع أو اسم قسم .

مثال ٧ - ٢٨ :

```
PERFORM SAMPLE-PARAGRAPH
COMPUTE ...
.....
SAMPLE-PARAGRAPH.
  ADD ...
  MOVE ...
  SUBTRACT ...

NEXT-PARAGRAPH.
  OPEN ...
  MOVE ...
```

تتسبب عبارة PERFORM في تنفيذ كل العبارات الموجودة في SAMPLE - PARAGRAPH في صورة متتابعة معتادة .. بعد تنفيذ SUBTRACT .. ينتقل الكمبيوتر إلى العبارة التي تلي عبارة PERFORM مباشرة .
عندما ينفذ قسم (والذي يمكن أن يحتوي على عديد من المقاطع) ، فتنفذ العبارات بدءاً بأول عبارة ، في أول مقطع وينتهي التنفيذ بأخر عبارة في آخر مقطع.

مثال ٧ - ٢٩ :

بمعرفة المقطع المعرف في مثال ٧ - ٢٨ تتسبب عبارة التنفيذ التالية :

```
PERFORM SAMPLE-PARAGRAPH THRU NEXT-PARAGRAPH
COMPUTE ...
.....
```

في أن يبدأ الكمبيوتر تنفيذ العبارات الموجودة في SAMPLE - PARAGRAPH في صورة متتابعة ، بدءاً بعبارة ADD يستمر التنفيذ ، عبارة بعد أخرى ، ومقطعاً تلو الآخر ، حتى تنفذ آخر عبارة في مقطع THRU (عبارة MOVE) ،

وعندها يستمر تنفيذ الكمبيوتر للعبارة في تتابع معتاد ، بدءا بالعبارة التي تلى عبارة **PERFORM** مباشرة .
(**COMPUTE**) .

عندما يستخدم جزء **THRU** مع أقسام .. يبدأ التنفيذ بأول عبارة (فى أول مقطع) من أول قسم ، وينتهى بأخر عبارة (فى آخر مقطع) من آخر قسم .

وبصفة عامة .. فإنه من الأفضل تجنب استخدام جزء **THRU** فى أى صيغة من صيغ **PERFORM** ؛ حيث إن تنفيذ **... THRU .. PERFORM** يعتمد على الموقع الطبيعي فى برنامج المصدر ؛ إذ توجد إمكانية بأنه يدخل مبرمج الصيانة مقطعا أو قسما جديدا فى مدى **THRU .. PERFORM** عن طريق الخطأ . كما أن استخدام **THRU .. PERFORM** يجعل برنامج المصدر أكثر صعوبة فى فهمه ، حيث يجب أن تذكر مواقع بداية ونهاية **PERFORM** . وهناك مواقف قليلة نسبيا فى برمجة الكوبل ، تدعو الحاجة فيها إلى استخدام **THRU .. PERFORM** ، وعلى هذا فيجب ألا يكون تجنبها مرفقا .

صيغة عدد من المرات

الصيغة التالية صيغة لأبسط من صيغ عبارة التنفيذ هي صيغة عدد من المرات **TIMES** ، والتي لها الشكل التالى :

$$\text{PERFORM procedure-name-1} \left[\left\{ \begin{array}{c} \text{THROUGH} \\ \text{THRU} \end{array} \right\} \text{procedure-name-2} \right] \left\{ \begin{array}{c} \text{identifier-1} \\ \text{integer-1} \end{array} \right\} \text{TIMES}$$

يلعب جزء **THRU** نفس الدور تماما فى تعريفه مدى من العبارات التى تنفذ ، كما يفعل فى عبارة التنفيذ البسيطة . مرة أخرى .. لا نوصى باستخدامه . عندما يستخدم **identifier-1** فيجب أن يكون عنصراً فردياً وصحيحاً (مثل **S9 PIC** بدون علامة عشرية) . يحدد **integer-1** أو **identifier-1** عدد المرات التى يجب أن ينفذ فيها المدى المحدد للعبارة ، قبل أن ينتقل الكمبيوتر للعبارة التى تلى عبارة **PERFORM** مباشرة . إذا كان **integer-1** أو **identifier-1** سالباً أو صفراً ، فلا ينفذ المدى المحدد للعبارة على الإطلاق ، وينتقل الكمبيوتر إلى العبارة التالية فوراً .

مثال ٧ - ٢٠ :

05 WS:NUMBER-OF-TEMPLATES PIC 99.

ACCEPT WS:NUMBER-OF-TEMPLATES FROM OPERATOR-OPTIONS-DEVICE

PERFORM PRINT-CHECK-ALIGNMENT-TEMPLATE
WS:NUMBER-OF-TEMPLATES TIMES

هنا .. يستخدم عنصر بيانات عددي فى تحديد العدد المطلوب التكرار . يجب أن يكون **OPERATOR - OPTIONS** معرفاً فى مقطع الأسماء الخاصه من جزء الأوساط . لاحظ انه إذا أدخل مشغل الكمبيوتر صفراً فى **WS - DEVICE** ، فلا تنفذ عبارة **PERFORM** المقطع **PRINT - CHECK - ALIGNMENT - NUMBER - OF - TEMPLATES** ، على الإطلاق .

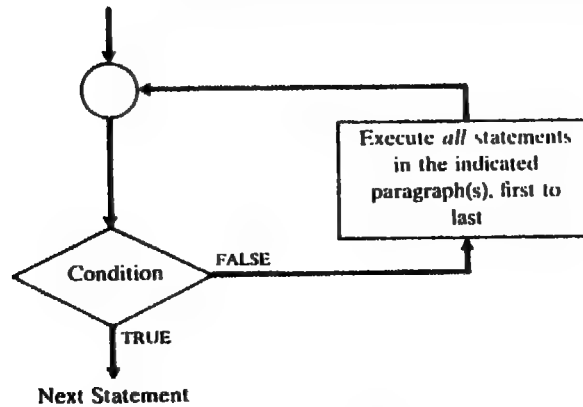
بالرغم من بساطة `PERFORM ... TIMES` إلا أنها لا تستخدم بكثرة في برامج الكويل : نظرا لأنه من النادر أن يعرف عدم مرات التكرار مسبقا . الأكثر اعتيادا هو حاجتنا الى تكرار تنفيذ أحد الأجزاء حتى يتحقق شرط معين ، يحدد أن التكرار يتوقف (مثل نهاية الملف) .

صيغة حتى

إن تكوين صيغة حتى `UNTIL` كما يلي :

`PERFORM` procedure-name-1 [`THROUGH` / `THRU`] procedure-name-2 `UNTIL` condition-1

يوجد تأثير .. `PERFORM ... UNTIL` في خريطة المسار المرتبة في شكل ٧ - ١١ .



شكل (٧ - ١١)

بالمقارنة مع شكل ٧ - ٥ : فإننا نميز هيكل `DO WHILE` (مع شرط نفى) . وعلى هذا .. كل الخواص ، التي سبق الإشارة لها في مثال ٧ - ٣ تسرى على `PERFORM ... UNTIL` . ومرة أخرى .. لا نوصى باستخدام جزء `THRU` .

مثال ٧ - ٣١ :

```

MOVE "NO" TO NO-MORE-SWITCH
PERFORM PRINT-A-MESSAGE
UNTIL NO-MORE-SWITCH EQUAL "YES"
.....
PRINT-A-MESSAGE.
WRITE PRINT-LINE
FROM WS-MESSAGE-LINE
    
```


مثال ٧ - ٢٢ :

```

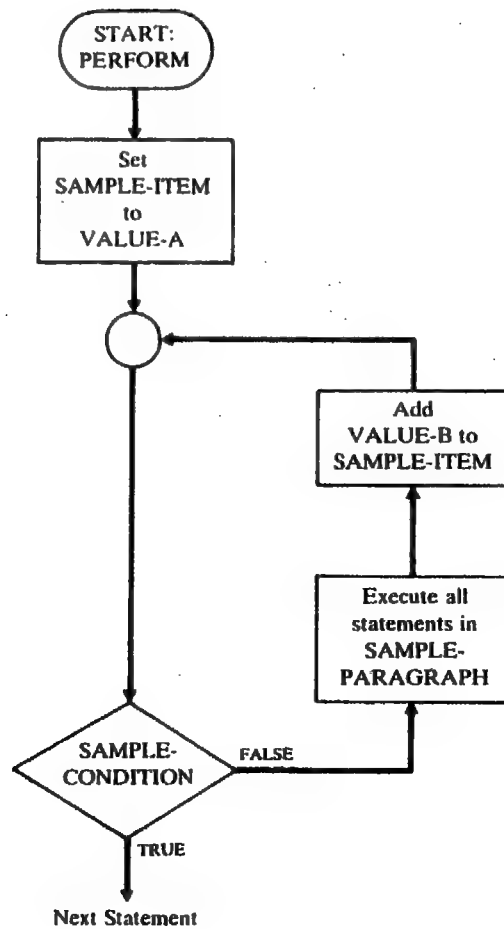
PERFORM ASSIGN-MAILBOXES
  VARYING MAILBOX-NUMBER FROM 1 BY 1
  UNTIL ALL-EMPLOYEES-PROCESSED
.....
ASSIGN-MAILBOXES.
  READ EMPLOYEE-MASTER-FILE
  INTO WS-EMPLOYEE-RECORD
  AT END
    MOVE "YES" TO ALL-EMPLOYEES-PROCESSED-SW

  IF ANOTHER-EMPLOYEE-INPUT
    MOVE MAILBOX-NUMBER TO WS-OUTPUT-MAILBOX-NUMBER
    MOVE WS-EMPLOYEE-NAME TO WS-OUTPUT-NAME
    WRITE MAILBOX-LIST-LINE
    FROM WS-OUTPUT-LINE

```

هذا الجزء مصمم لتحديد صناديق بريد للشركة (مرقمة 1 , 2 , 3 , ...) وذلك للعاملين بالشركة . يستخدم جزء VARYING من عبارة PERFORM في التحكم في رقم صندوق البريد MAILBOX - NUMBER الذي يحدد للعامل الموجودة بياناته تحت التشغيل الآن . وتوضع قيمة ابتدائية للعنصر MAILBOX - NUMBER من قيمة جزء FROM وهي 1 . يتم تقويم شرط UNTIL عند ذلك .. فإذا كان خطأ فينفذ المقطع ASSIGN - MAILBOXES . يقرأ مقطع ASSIGN - MAILBOXES سجل أحد العاملين ، فإذا لم يصل الى نهاية الملف ، فيطبع سطرا مبيئا فيه اسم العامل والمحتويات الحالية للعنصر MAILBOX - NUMBER . بعد تنفيذ آخر عبارة في المقطع ASSIGN - MAILBOXES تضاف قيمة BY ، وهي 1 ، إلى MAILBOX - NUMBER لتزيد محتوياته وتصبح 2 ، عند ذلك .. يعاد تقويم شرط UNTIL ... إلخ . بعد اتمام التشغيل لآخر عامل (وتحديد رقم صندوق بريد n له ، إذا كان عدد العاملين هو n) ، يصبح الشرط ALL - EMPLOYEES-PROCESSED صحيحا (YES) ، ويقفز الكمبيوتر إلى أول عبارة تلى عبارة PERFORM .

حدث أن تساوت قيمة FROM مع قيمة BY ، في المثال . يبين شكل ١٣ - ٧ خريطة المسار المرتبة المناظرة لعبارة PERFORM الأكثر عمومية التالية :



شكل (٧ - ١٣)

مثال ٧ - ٢٢ :

يمكن كتابة عبارة PERFORM التالية :

PERFORM SAMPLE - PARA 10 TIMES

كما يلي :

PERFORM SAMPLE-PARA
 VARYING NUMBER-OF-TIMES FROM 1 BY 1
 UNTIL NUMBER-OF-TIMES GREATER THAN 10

، أو كما يلي :

PERFORM SAMPLE-PARA
 VARYING NUMBER-OF-TIMES FROM 1 BY 1
 UNTIL NUMBER-OF-TIMES EQUAL 11

من الحرج أن يفهم أن ما يلي :

```
PERFORM SAMPLE-PARA
  VARYING NUMBER-OF-TIMES FROM 1 BY 1
  UNTIL NUMBER-OF-TIMES IS EQUAL TO 10
```

ينتج عنه تنفيذ SAMPLE - PARA تسع مرات فقط . السبب موضح ببساطة في شكل ٧ - ١٣ . وبالمثل تتسبب :

```
PERFORM COMPARE-LOCATION-CODES
  VARYING LOCATION-NUMBER FROM 1 BY 1
  UNTIL LOCATION-NUMBER EQUAL TO MAXIMUM-LOCATIONS
```

في تنفيذ COMPARE - LOCATION - CODES عدد من المرات ، أقل مرة واحدة عن المحتويات الحالية في MAXIMUM - LOCATIONS .

مثال ٧ - ٢٤ :

```
PERFORM COMPARE-CODES
  VARYING NUMBER-COMPARES FROM VALUE-A BY VALUE-B
  UNTIL NUMBER-COMPARES GREATER THAN VALUE-C
```

لأي تغييرات في NUMBER - COMPARES ، أو في VALUE - B أو في VALUE - C حدثت داخل المقطع COMPARE - CODES نفس التأثير المتوقع على عدد مرات التكرار . إلا أن تغيير قيمة FROM (وهي VALUE - A) أثناء تنفيذ PERFORM لا يؤثر على عدد مرات التكرار ، حيث تلعب قيمة FROM دوراً في بداية تنفيذ PERFORM فقط (أى إنها تضع القيمة الابتدائية لمحتويات عنصر بيانات VARYING) .

مثال ٧ - ٢٥ :

```
PERFORM SAMPLE-PARA
  VARYING ITEM-A FROM ITEM-B BY ITEM-C
  UNTIL ITEM-A GREATER THAN ITEM-D
```

عندما تستخدم عناصر بيانات مع PERFORM ... VARYING ... يجب أن تكون كلها عناصر بيانات فردية عددية . وعلى هذا .. يجب أن تكون صورة العناصر ITEM-A ، ITEM-B ، ITEM-C ، ITEM-D بها 9 ، ويمكن أن يكون بها S ، أو اختياريين .

```
PERFORM SAMPLE-PARA
  VARYING ITEM-A FROM .25 BY .03
  UNTIL ITEM-A GREATER THAN .40
```


ينفذ SAMPLE - PARA عدد 6 مرات ، مع أخذ العنصر ITEM-A القيم 25, 28, 31, 34, 37, 40, ، وينتهي تنفيذ PERFORM عندما تساوى قيمة ITEM-A 43.

- PERFORM SAMPLE-PARA
VARYING ITEM-A FROM .25 BY .03
UNTIL ITEM-A EQUAL TO .41

ينتج عن عبارة PERFORM هذه دورة لا نهائية ، حيث لا يتحقق الشرط ITEM-A EQUAL TO .41 على الإطلاق (الرقم 25 - 41 لا يقبل القسمة على 03) . انظر مثال ٧ - ٣١

- PERFORM SAMPLE-PARA
VARYING ITEM-A FROM 10 BY -1
UNTIL ITEM-A IS ZERO

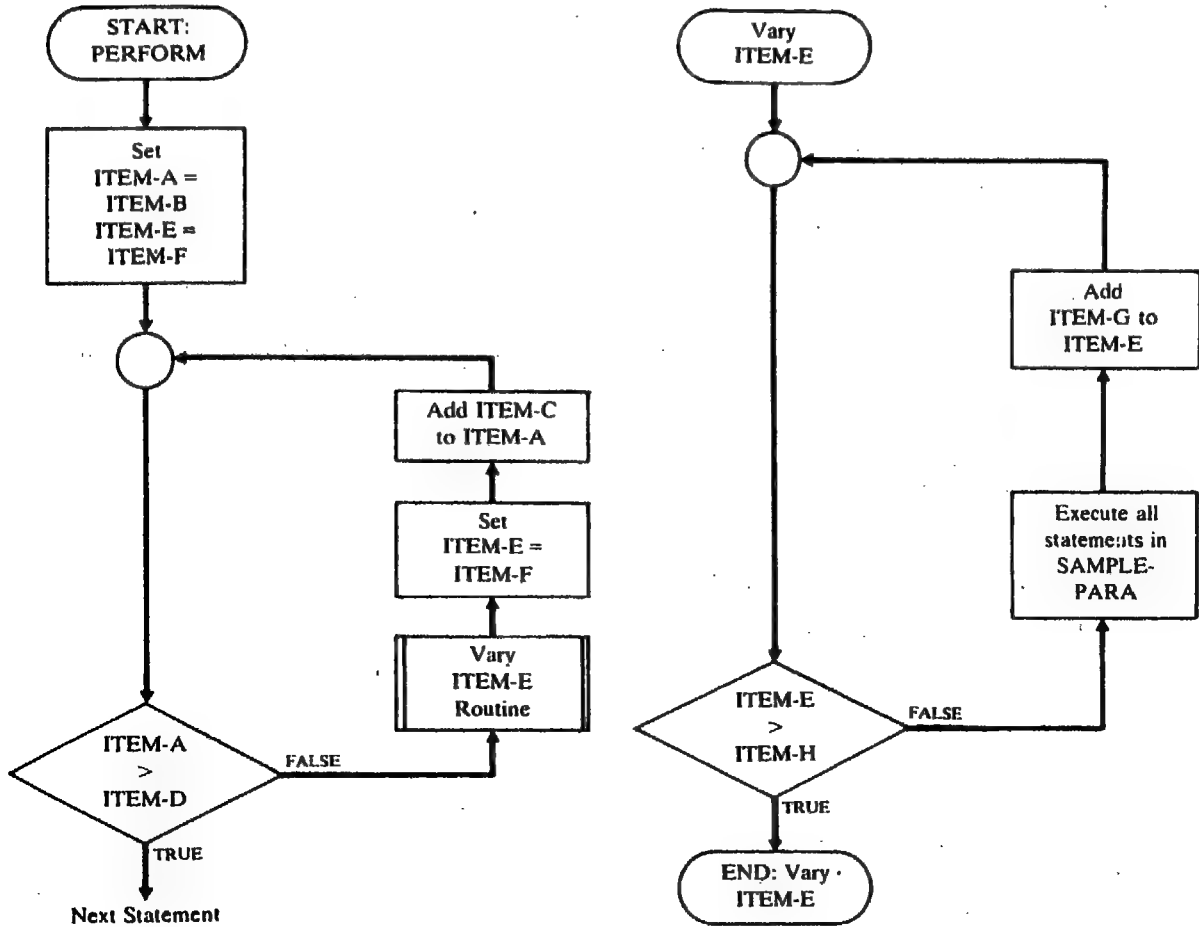
عندما تكون قيمة BY سالبة .. فيأخذ العنصر ITEM-A القيم التالية : 1,2,3,4,5,6,7,8,9,10 ، وينفذ SAMPLE - PARA عشر مرات . بعد التنفيذ للمرة العاشرة .. يساوى ITEM - A صفراً ، وينتقل الكمبيوتر الى العبارة التى تلى عبارة PERFORM مباشرة .

عندما يتغير عنصرا بيانات فى عبارة PERFORM...VARYING .

PERFORM SAMPLE-PARA
VARYING ITEM-A FROM ITEM-B BY ITEM-C
UNTIL ITEM-A GREATER THAN ITEM-D
AFTER ITEM-E FROM ITEM-F BY ITEM-G
UNTIL ITEM-E GREATER THAN ITEM-H

فيكون التنفيذ كما يبدو فى خريطة المسار المرتبة فى شكل ٧ - ١٤ . يمكن رؤيته أنه لكل قيمة من قيم ITEM - A التى تقع فى المدى (ITEM - B ≤ ITEM - A ≤ ITEM - D) ينفذ المقطع SAMPLE - PARA لكل قيم ITEM - E (التى تقع فى المدى (ITEM - F ≤ ITEM - E ≤ ITEM - H) . وعلى هذا .. يكون تأثير PERFORM ... VARYING هو تنفيذ متكرر للمقطع SAMPLE - PARA مع :

إجمالى عدد مرات التكرار = (عدد القيم التى تقع فى مدى ITEM - A) x (عدد القيم التى تقع فى مدى ITEM - E).



شكل (٧ - ١٤)

مثال ٧ - ٣٦ :

PERFORM CALCULATE-MORTGAGE-PAYMENT
 VARYING PAYMENT-YEAR FROM 83 BY 1
 UNTIL PAYMENT-YEAR GREATER THAN 93
 AFTER PAYMENT-MONTH FROM 1 BY 1
 UNTIL PAYMENT-MONTH GREATER THAN 12

ينفذ CALCULATE - MORTGAGE - PAYMENT لكل قيمة من قيم PAYMENT - YEAR من 83 إلى 93 ، مع شمول الحدود . ولكل من هذه القيم البالغ عددها 11 للعنصر PAYMENT - YEAR تتغير قيمة PAYMENT - MONTH من 1 إلى 12 . وعلى هذا .. يحسب المقطع (11x12) 132 قيمة لدفعات القرض ، والتي يمكن أن تمثل على هيئة جدول ، أو في صورة خطية (شكل ٧ - ١٥) .

Month \ Year	1	2	...	12
83	\$xxx			
84				
...				
93				

(a)

Year	Month	Payment
83	1	\$xxx
83	2	
...
83	12	
84	1	
84	2	
...
84	12	
...
93	1	
93	2	
...
93	12	

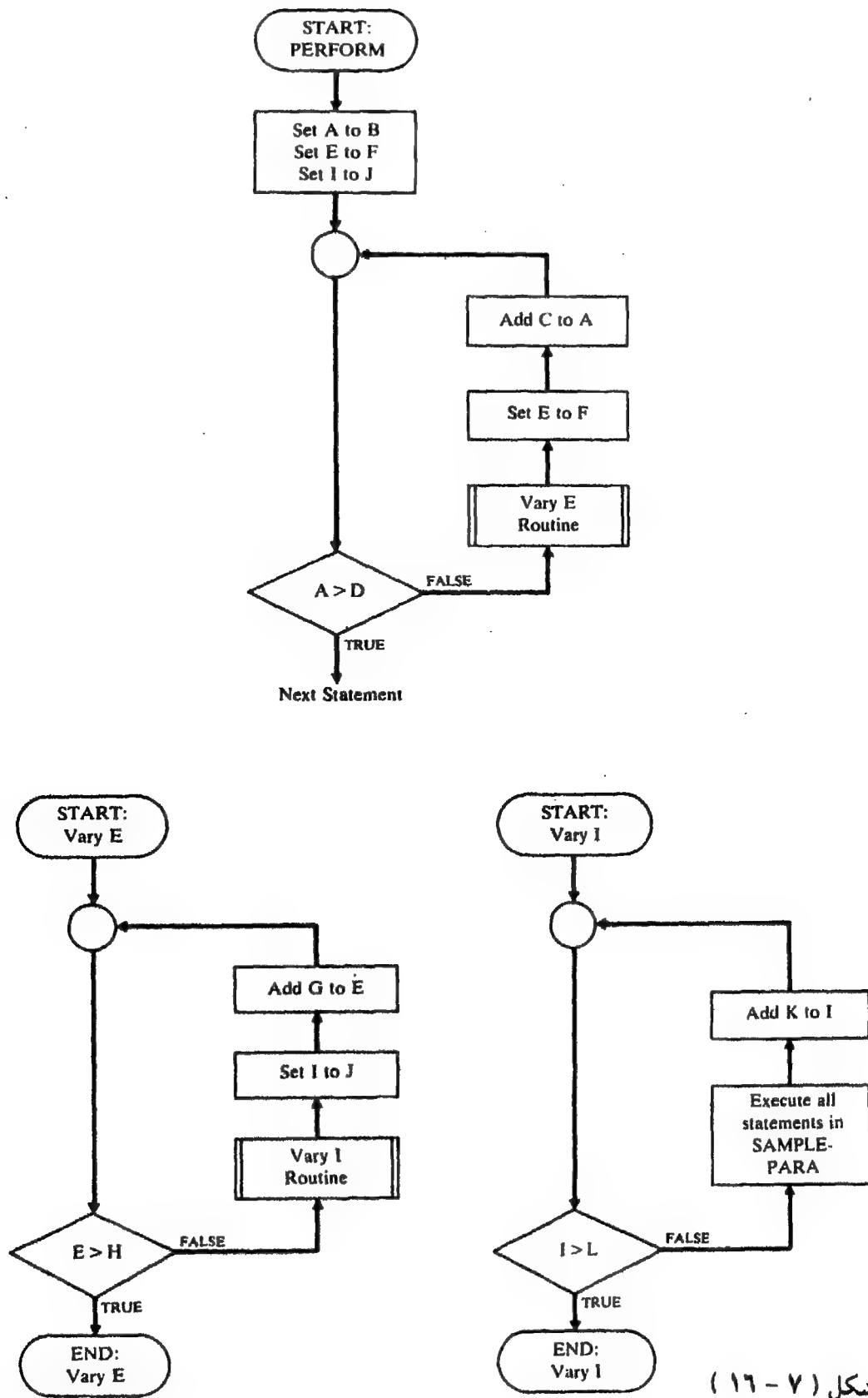
(b)

شكل (٧ - ١٥)

يمكن أن يتغير أقصى رقم لعناصر بيانات 3 في عبارة PERFORM ... VARYING (انظر ملحق حـ لمعرفة العدد المناظر في كويل 1985 النمطى) :

```
PERFORM SAMPLE-PARA
  VARYING A FROM B BY C
    UNTIL A GREATER THAN D
  AFTER E FROM F BY G
    UNTIL E GREATER THAN H
  AFTER I FROM J BY K
    UNTIL I GREATER THAN L
```

لكل قيمة تؤخذ للعنصر A .. تتغير قيم E خلال مداها كله ، ولكل قيمة من قيم E ، تتغير قيم I خلال مداها كله . يتبع ذلك أن عدد مرات تكرار المقطع SAMPLE - PARA هو : (عدد قيم A) x (عدد قيم E) x (عدد قيم I) ، ويوضح شكل (٧-١٦) خريطة المسار المرتبة لهذه الحالة .



شكل (٧ - ١٦)

مثال ٧ - ٢٧ :

```

PERFORM PRODUCE-TOTAL-INTEREST
  VARYING MORTGAGE-AMOUNT
    FROM 50000 BY 1000
    UNTIL MORTGAGE-AMOUNT GREATER THAN 100000
  AFTER YEARS-TO-REPAY
    FROM 10 BY 1
    UNTIL YEARS-TO-REPAY GREATER THAN 30
  AFTER INTEREST-RATE
    FROM .08 BY .01
    UNTIL INTEREST-RATE GREATER THAN .20

```

يمكن استخدام عبارة PERFORM في إنتاج تقرير يبين إجمالي قيم الفائدة لقيم القرض المختلفة ، عند معدلات فائدة مختلفة بأطول زمنية مختلفة للسداد . يمكن أن تتغير قيم القرض من 50,000 إلى 100,000 بخطوات متساوية تساوى 1000 . لكل قيمة من قيم MORTGAGE - AMOUNT .. تتغير قيمة YEARS - TO - REPAY من 10 إلى 30 بخطوات متساوية تساوى 1 . لكل قيمة من قيم MORTGAGE - AMOUNT ، وكل قيمة من قيم YEARS - TO - REPAY تتغير قيمة INTER - RATE من 8% إلى 20% بخطوات متساوية تساوى 1% . يمكن طباعه التقرير كقائمة خطية ، أو ربما من الأفضل ، كمجموعة من 51 جنولا ، يعطى كل منها إجمالي الفائدة لكل خليط لأطول الفترة ومعدل الفائدة .

٧ - ١٣ عبارات التنفيذ المتداخلة

كما أنه يسمح صحيح استخدام عبارة IF داخل عبارته IF (القسم رقم 11 من هذا الفصل) .. فمن الصحيح كذلك استخدام عبارة PERFORM داخل مقطع ينفذ بواسطة عبارة PERFORM أخرى .

مثال ٧ - ٢٨ :

```

PERFORM PARA-A
MOVE "YES" TO SAMPLE-SWITCH
.....
PARA-A.
  MOVE 1 TO L
  MOVE 2 TO M
  PERFORM PARA-B
  MOVE 3 TO N
  .....
PARA-B.
  ADD 1 TO L
  ADD 2 TO M
  PERFORM PARA-C
    VARYING AN-ITEM FROM 1 BY 1
    UNTIL AN-ITEM GREATER THAN 3
  ADD 3 TO P
  .....
PARA-C.
  MOVE SPACES TO X
  MOVE LOW-VALUES TO Y

```

يتسبب تنفيذ عبارة PERFORM في أعلى مستوى في تنفيذ PARA - A مرة واحدة . إلا أن المقطع PARA - A يحتوى على عبارة PERFORM ؛ بحيث يؤدي تنفيذ PARA - A إلى تنفيذ للمقطع PARA - B مرة واحدة . بالمثل يحتوى المقطع PARA - B على عبارة PERFORM ، بحيث أن تنفيذ PARA - B ينتج عنه تنفيذ PARA - C ثلاث مرات . تذكر أنه عندما ينتهى تنفيذ عبارة PERFORM .. يعود التحكم تلقائياً إلى العبارة التى تلى عبارة PERFORM مباشرة . وعلى هذا ينتقل الكمبيوتر تلقائياً من ثالث تكرار لآخر عبارة فى PARA - C إلى العبارة التى تلى عبارة PERFORM فى PARA - B . من آخر عبارة فى PARA - B .. ينتقل الكمبيوتر تلقائياً إلى العبارة التى تلى عبارة PERFORM فى PARA - A ومن آخر عبارة فى PARA - A .. ينتقل الكمبيوتر إلى العبارة التى تلى عبارة PERFORM التى بدأت كل هذا التسلسل . وعلى هذا تنفذ العبارات بالترتيب التالى :

1. PERFORM PARA-A
2. MOVE 1 TO L
3. MOVE 2 TO M
4. PERFORM PARA-B
5. ADD 1 TO L
6. ADD 2 TO M
7. PERFORM PARA-C VARYING ...
8. MOVE SPACES TO X (1st iteration; AN-ITEM = 1)
9. MOVE LOW-VALUES TO Y (1st iteration; AN-ITEM = 1)
10. MOVE SPACES TO X (2nd iteration; AN-ITEM = 2)
11. MOVE LOW-VALUES TO Y (2nd iteration; AN-ITEM = 2)
12. MOVE SPACES TO X (3rd iteration; AN-ITEM = 3)
13. MOVE LOW-VALUES TO Y (3rd iteration; AN-ITEM = 3)
14. ADD 3 TO P
15. MOVE 3 TO N
16. MOVE "YES" TO SAMPLE-SWITCH

يشمل المصدر الرئيسى للخطأ عند استخدام عبارات PERFORM متداخلة كتابة عبارة PERFORM ، التى تنفذ نفسها بصورة مباشرة أو غير مباشرة . عندما يستدعى جزء من البرنامج .. فيقال عنه أنه يتسم بالإعاده الذاتية recursive . الإعاده الذاتية لعبارات PERFORM - غير مسموح بها فى الكويل (بالرغم من أن الإعاده الذاتية ممكنة فى لغات أخرى للبرمجه مثل PL/I) .

مثال ٧ - ٢٩ :

```

PROCEDURE DIVISION.
  OPEN INPUT TEST-FILE
  MOVE "NO" TO END-FILE-SWITCH
  PERFORM READ-AND-VALIDATE
    UNTIL END-FILE-SWITCH EQUAL "YES"
  CLOSE TEST-FILE
  STOP RUN

READ-AND-VALIDATE.
  READ TEST-FILE
  AT END
    MOVE "YES" TO END-FILE-SWITCH

  IF END-FILE-SWITCH EQUAL "NO"
    IF TEST-RECORD-FIELD NUMERIC
      PERFORM VALID-RECORD-PROCESSING
    ELSE
      PERFORM PRODUCE-ERROR-MESSAGE
      PERFORM READ-AND-VALIDATE

```

آخر عبارة في المقطع READ - AND - VALIDATE هي عبارة PERFORM READ - AND - VALIDATE. وعلى هذا .. تحت الظروف الصحيحة (TEST - RECORD - FIELD ليس عددياً) تنفذ عبارة PERFORM نفسها. هذا خطأ برمجة بالقطع، يبدو في كويل IBM OS/VS كدورة لا نهائية.

يفكر المبرمج حتماً أنه إذا ما وجد سجلاً غير صحيح .. فإن الشيء المناسب عمله هو طباعة رسالة خطأ، والاستمرار بقراءة السجل التالي وتشغيله. الطريقة صحيحة في الواقع - ولكن ما سبق أن نسيه هو أن جزء READ - AND - VALIDATE يقع تحت تحكم عبارة PERFORM...UNTIL سابقة. ليس من الصحيح تنفيذ PERFORM ثانية للمقطع، قبل اتمام تنفيذ PERFORM الأولى. في هذه الحالة، تجعل PERFORM...UNTIL الأولى من PERFORM الثانية عبارة غير ضرورية بأي حال من الأحوال.

مثال ٧ - ٤٠ :

```

PROCEDURE DIVISION.
.....
PARA-A.
  MOVE 1 TO L
  MOVE 2 TO M
  PERFORM PARA-B
  MOVE 3 TO N
.....
PARA-B.
  ADD 1 TO L
  ADD 2 TO M
  PERFORM PARA-C
  ADD 3 TO P

```

PARA-C.
MOVE SPACES TO X
PERFORM PARA-A
MOVE LOW-VALUES TO Y

هنا تدور العبارة B - PARA - B حول تنفيذ نفسها ، ولكن هذا بطريقة غير مباشرة ، وذلك بعد سلسلة من عبارات PERFORM متداخلة . وهذا خطأ يشبه الدورة اللانهائية . يحتوى A - PARA على عبارة PERFORM تنفذ PARA B - (هذا صحيح حتى الآن) . ويحتوى B - PARA على عبارة PERFORM تنفذ C - PARA (مازال هذا صحيحاً) ، إلا أن C - PARA يحتوى على عبارة PERFORM تنفذ A - PARA ، وهذا غير صحيح لأنه أثناء إعادة تنفيذ A - PARA .. تنفذ العبارة B - PARA ، بينما لا تكون عبارة B - PARA قد نفذت بالفعل.

واتجنب مثل هذه المواقف الموجودة في مثال ٧ - ٤٠ فإننا نلاحظ ما يلي :

قاعدة لعبارات PERFORMs المتداخلة ..

المقطع أو القسم الذى ينفذ تحت تحكم عبارة PERFORM يجب ألا ينفذ مرة أخرى ، حتى ينتهى تنفيذ عبارة PERFORM المتحكمه فيه أولا .

عندما يعود التحكم من الإجراء المنفذ .. فسيكون مسموحاً ، بالطبع ، بتنفيذ الإجراء مرة أخرى .

٧ - ١٤ التصميم المنطقى : الشجرة الشبيهة

عادة ما تستخدم الشجرة الشبيهة pseudocode بدلا من خرائط المسار المرتبة .. ، وذلك فى تصميم منطق البرنامج وتوثيقه. ونظرا لتشابه الشجرة الشبيهة مع اللغة الانجليزية ، فعادة ما يشار إليها بالانجليزية المرتبة structured english . وحيث إن الشجرة الشبيهة ليست رسومات .. فهى أسهل كثيرا فى إنتاجها وتعديلها عن خرائط المسار المرتبة .. (التي يجب أن ترسم بعناية). ولا توجد صيغة موحدة للشجرة الشبيهة ، والمتطلب الوحيد هو أن الصيغة تكون قادرة على تمثيل الثلاثة هياكل الرئيسية للمنطق : تتابع واختيار وتكرار . توصف المدخلات والتشغيل والمخرجات للبيانات بعبارات إنجليزية (موجزة) ، ويستخدم الترحيل فى توضيح العلاقات بين عبارات الشجرة الشبيهة .

مثال ٧ - ٤١ :

شجرة شبيهة لهياكل المنطق الأساسية .

Sequence

increment days overdue counter by 1
compute late charge as 5% of balance due
build dunning statement for printing

Selection

if hours worked is greater than 40
.....
else
.....
endif

Iteration

perform until transaction customer ID not equal master customer ID
.....
endperform

من الممكن بالطبع نهج مناهج أخرى ، طالما أنها (١) تفهم قراءتها (٢) ومتناسكة (أي إنك إذا استخدمت perform/ end- perform للتكرار في أحد الأماكن من الشفرة الشبيهة ، فعليك باستخدامها في كل المواقع كذلك) .

مثال ٧ - ٤٢ :

اذكر شفرة شبيهة مكافئة (أ) لشكل ٧ - ٤ (ب) لشكل ٧ - ٧

```
(i)
read payment date, payment, due date, old balance
if payment date greater than due date
    call late payment routine
else
    if payment greater than balance
        calculate refund
        format check
        print check
    else
        compute new balance as old balance - payment
endif
endif
build cash received line
```

لاحظ أن هذه الشفرة الشبيهة تستدعى عملية سبق تعريفها في العبارة : call payment routine . يمكن أن يقبل كذلك استخدام perform late payment routine و execute late payment routine ، أو أى عبارة تحدد بوضوح ما يراد عمله عند هذه النقطة من الخوارزمى .

```
(ب)
set end-cards to "no"
set #employees to zero
read card-file at end set end-cards to "yes"
perform until end-cards equal "yes"
    increment #employees by 1
    move name, regular hours, and overtime hours to output areas
    compute total-hours as regular hours + overtime hours
    print time listing line with name, regular hours, overtime hours, and
        total hours
    read card-file at end set end-cards to "yes"
endperform
print #employees
stop
```

وصيغة الشفرة الشبيهة أكثر إيجازاً عن خريطة المسار المرتبة . لاحظ أن عبارات الشفرة الشبيهة المراد تكرارها في ميكل تكرار ، مكتوبة بين عبارات perform until ، endperform في ترجمة الكويل ، العبارات المراد تكرارها بالطبع ، كمقطع منفصل ، ينفذ بواسطة PERFORM ... UNTIL (انظر ملحق حد بالنسبة إلى كويل 1985 النمطى) .

يفضل معظم المبرمجين الشفرة الشبيهة عن خرائط المسار المرتبة . وبأى طريقة .. يجب أن يصمم البرنامج قبل كتابة أى شفرة بلغة الكويل . يجب كتابة جزء الإجراءات في الواقع باستخدام الشفرة الشبيهة أو خريطة المسار المرتبة كدليل . يمكن توفير كثير من وقت التصحيح إذا ما اختبرت الشفرة الشبيهة أو خريطة المسار المرتبة قبل بدء كتابة شفرة الكويل . ويعنى الاختبار أن تقوم بالدور الذى يلعبه الكمبيوتر ، وتتبع الخوارزمى خطوة بخطوة ، مجرياً تشغيل لبعض بيانات اختيارية ومنتجا نتائج ؛ فإذا كانت النتائج صحيحة ، فيجب أن يكون الخوارزمى صحيحاً . وإذا ما تمت ترجمته ترجمة صحيحة إلى الكويل .. فيجب أن يكون برنامج الكويل صحيحاً .

٧ - ١٥ التصميم المنطقي : جداول القرارات

يمكن أن تصبح الشجرة الشبيهة وخرائط المسار المرتبة مرهقة عندما يتطلب الخوارزمي هياكل اختيار معقدة (مثل التداخل العميق). جدول القرار DECISION TABLE عبارة عن رسم يلخص بطريقة مقنعة إجراءات قرارات معقدة - complicated decision procedures ، والأكثر دقة ، أنه جدول حقيقي (القسم رقم ٩ من هذا الفصل) لمجموعة من العبارات (تسمى إجراءات actions المنطقية (المعقدة) والمركبة من مجموعة من العبارات البسيطة (تسمى شروطاً) ، وتخطيط جدول القرارات م - نصح في شكل ٧ - ١٧ .

condition stub	conditions
action stub	actions

شكل (٧ - ١٧)

مثال ٧ - ٤٣ :

شكل ٧ . ١٨ هو جدول قرارات يمثل شراء على الحساب من أحد المحلات . جزء الشروط condition stub يسرد ثلاثة شروط ، كل منها يمكن أن يكون صحيحاً (نعم Y) أو خطأ (لا N) .
لاحظ أن كل الخليط الممكن من الصحة والخطأ للثلاثة شروط ممثل بأعمدة شروط 2^3 أي ٨ أعمدة . (لعدد n من الشروط يصبح عدد الأعمدة 2^n) . جزء الإجراءات action stub يسرد ثلاثة إجراءات ممكنة يمكن اتخاذها بالنسبة للشراء . (بالصدفة تساوي عدد الإجراءات مع عدد الشروط هنا). في كل عمود توجد X عند كل إجراء يجب أخذه لخليط معين من قيم الحقيقة . انظر التمرين رقم ٩١ في هذا الفصل للشجرة الشبيهة لجدول القرارات هذا ، و جدول القرارات أسهل كثيراً في صياغته عن الشجرة الشبيهة).

Account Active	Y	Y	Y	Y	N	N	N	N
Last Month's Payment Received	Y	Y	N	N	Y	Y	N	N
Purchase Is Within Credit Limit	Y	N	Y	N	Y	N	Y	N
Approve Purchase	x	x	x					
Refer Customer to Credit Office		x	x					
Disapprove Purchase				x	x	x	x	x

شكل (٧ - ١٨)

أسئلة للمراجعة :

- ٧ - ١ ماذا يعنى منطق البرنامج ؟
- ٧ - ٢ ما الثلاث هياكل المنطقية الاساسية ؟
- ٧ - ٣ ما خريطة المسار المركبة ؟ وكيف تستخدم ؟
- ٧ - ٤ ما الغرض من كل رمز من رموز خريطة المسار المرتبة التي سبق ذكرها في هذا الفصل ؟
- ٧ - ٥ وضع كيف ترسم خريطة المسار لهيكل التتابع .
- ٧ - ٦ وضع كيف ترسم خريطة المسار لهيكل الاختيار .
- ٧ - ٧ وضع كيف ترسم خريطة المسار لهيكل التكرار .
- ٧ - ٨ وضع كيف يمكن دمج الثلاثة هياكل المناطق الثلاثة في خريطة مسار .
- ٧ - ٩ ما هيكل DO WHILE ؟
- ٧ - ١٠ كيف ينفذ هيكل الاختيار في الكويل ؟
- ٧ - ١١ كيف يمكن استخدام NEXT SENTENCE ؟
- ٧ - ١٢ وضع أهميه النقطة في عبارة IF .
- ٧ - ١٣ وضع تكوين واستخدام شرط الفته .
- ٧ - ١٤ متى يكون عنصر بيانات عدديا ؟ ومتى يكون حرفيا ؟
- ٧ - ١٥ أى عناصر البيانات يجب أن يتأكد برنامج الكويل من صحتها ؟
- ٧ - ١٦ ماذا يحدث إذا استخدمت البيانات عدديّة غير صحيحة في حساب أو في شرط علاقة ؟
- ٧ - ١٧ متى يجب ألا تختبر صحة بيانات المدخلات ؟
- ٧ - ١٨ وضع تكوين واستخدام شرط العلاقة .
- ٧ - ١٩ كيف تنفذ العلاقات «أقل من أو يساوى» و «أكبر من أو يساوى» في الكويل ؟
- ٧ - ٢٠ وضع القواعد المتبعة في مقارنة عنصرين عدديين في الكويل .
- ٧ - ٢١ كيف تقارن العناصر غير العددية في الكويل ؟
- ٧ - ٢٢ ما مجموعات خليط عناصر البيانات غير المسموح بها في شروط العلاقات ؟
- ٧ - ٢٣ وضع تكوين واستخدام شرط الإشارة .
- ٧ - ٢٤ اذكر ثلاث مميزات لاستخدام شروط أسماء شرطية بدلا من شروط العلاقات .
- ٧ - ٢٥ كيف تعرف الأسماء الشرطية ؟
- ٧ - ٢٦ كيف يتم تقويم شرط مركب ، مكون بواسطة المؤثرات العلاقية ؟ NOT, OR, AND ؟
- ٧ - ٢٧ متى يجب استخدام الأقواس في الشروط المركبة ؟

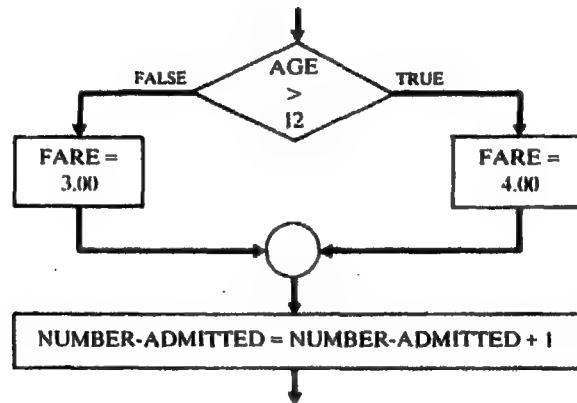
- ٧ - ٢٨ ما القواعد المتبعة إذا لم توجد أقواس في الشرط المركب ؟
- ٧ - ٢٩ اذكر قواعد اختصار شروط العلاقات المركبة . متى يجب استخدام الاختصار ؟
- ومتى لا يجب ؟
- ٧ - ٣٠ ماذا تعني IF المتداخلة ؟
- ٧ - ٣١ وضع كيف تتزاوج IF مع ELSE في IF المتداخلة .
- ٧ - ٣٢ وضع دور الترحيل عند كتابة IF متداخلة .
- ٧ - ٣٣ وضع استخدام NEXT SENTENCE في IF المتداخلة .
- ٧ - ٣٤ وضع هيكل الحالة .
- ٧ - ٣٥ ما هي IF الخطية ؟ وضع كيفية استخدامها في تنفيذ هيكل الحالة .
- ٧ - ٣٦ ناقش تكوينات الصيغ الأربعة لعبارة PERFORM .
- ٧ - ٣٧ وضع جزء THRU من عبارة PERFORM . لماذا يجب تجنب استخدامه ؟
- ٧ - ٣٨ كيف تستخدم PERFORM ... UNTIL في تنفيذ هيكل التكرار ؟
- ٧ - ٣٩ ماذا يحدث إذا أصبح شرط UNTIL صحيحا في منتصف تنفيذ المقطع ؟
- ٧ - ٤٠ ما الدورة اللانهائية ؟
- ٧ - ٤١ هل من الممكن تنفيذ صفر من المرات ؟ كيف ؟
- ٧ - ٤٢ ناقش الاختلافات بين : PERFORM ... VARYING ... AFTER ... PER- و PERFORM ... VARYING ... AFTER ... FORM .. VARYING ...
- ٧ - ٤٣ ناقش الاختلاف بين EQUAL, GREATER THAN بالنسبة الى عدد مرات التكرار التي تتحقق بواسطة PER- FORM ... UNTIL .
- ٧ - ٤٤ ماذا تعني عبارة PERFORM المتداخلة ؟
- ٧ - ٤٥ اذكر الخطأ الذي يمكن أن يقع من استخدام PERFORM المتداخلة .
- ٧ - ٤٦ ما الشفرة الشبيهة ؟ ولماذا تفضل عادة عن خريطة المسار المرتبة ؟
- ٧ - ٤٧ أكتب شفرة شبيهة . وارسم خريطة مسار لكل عبارة PERFORM في السؤال رقم ١٢ .
- ٧ - ٤٨ ما الاختبار الذي يجرى على الشفرة أو خريطة المسار ؟
- ٧ - ٤٩ وضع استخدام جداول القرارات . كيف ترسم هذه الجداول ؟

مسائل محلولة

٧ - ٥٠ أكتب شفرة كويل لشكل ٧ - ١٩

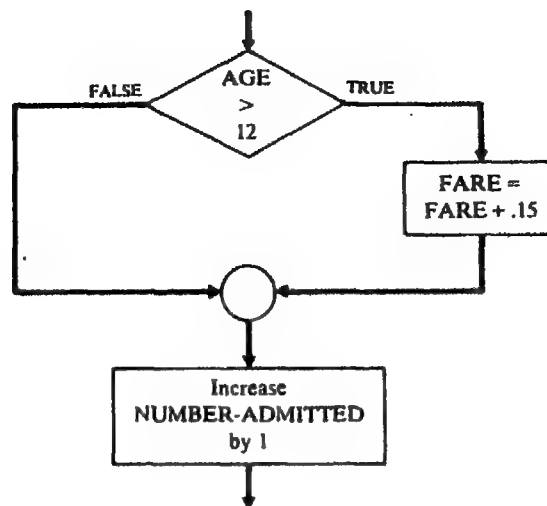
```
IF AGE IS GREATER THAN 12
  MOVE 4.00 TO FARE
ELSE
  MOVE 3.00 TO FARE
```

```
ADD 1 TO NUMBER-ADMITTED
```



شكل (٧-١٩)

٧-١٥ اكتب شفرة كويل لشكل (٧-٢٠)



شكل (٧-٢٠)

يكتب معظم مبرمجي الكويل ما يلي :

IF AGE GREATER THAN 12
ADD .15 TO FARE

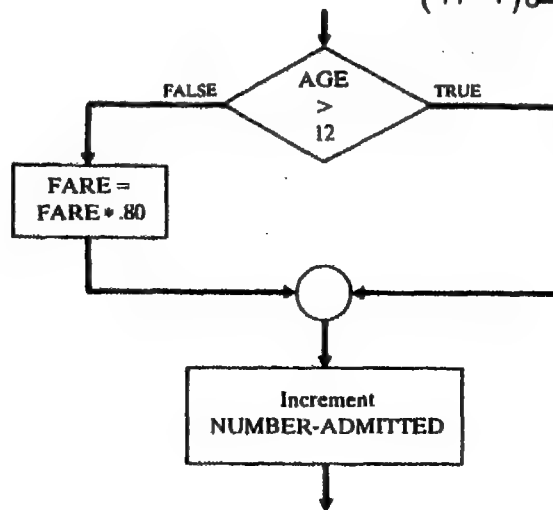
ADD 1 TO NUMBER-ADMITTED

إلا أن البعض يفضل وجود ELSE لكل IF :

```
IF AGE GREATER THAN 12
  ADD .15 TO FARE
ELSE
  NEXT SENTENCE
```

ADD 1 TO NUMBER-ADMITTED

٧-٢ هـ اكتب شفرة كويل لشكل (٧-٢١)



شكل (٧-٢١)

If NEXT SENTENCE is used, the solution is straightforward:

```
IF AGE GREATER THAN 12
  NEXT SENTENCE
ELSE
  COMPUTE FARE = FARE * .80
```

ADD 1 TO NUMBER-ADMITTED

قد يرى بعض المبرمجين أنه من الأفضل حذف NEXT SENTENCE بنفى الشرط في شكل ٧-٢١ ، وتبادل TRUE مع FALSE . وهذا يعطى الشفرة المكافئة التالية :

```
IF AGE NOT GREATER THAN 12
  COMPUTE FARE = FARE * .80
```

ADD 1 TO NUMBER-ADMITTED

٧ - ٥٣ حدد الخطأ فيما يلي :

```

IF CUSTOMER-TYPE EQUAL "3"
  MOVE ZERO TO SHIPPING-DISCOUNT
ELSE
  PERFORM DETERMINE-SHIPPING-DISCOUNT
  ADD SHIPPING-DISCOUNT TO TOTAL-DISCOUNT
  PERFORM FORMAT-INVOICE

```

إذا حدد الترحيل التشغيل المطلوب بطريقة صحيحة .. فستكون هناك حاجة إلى نقطة لإنهاء عبارة IF كما يلي :

```

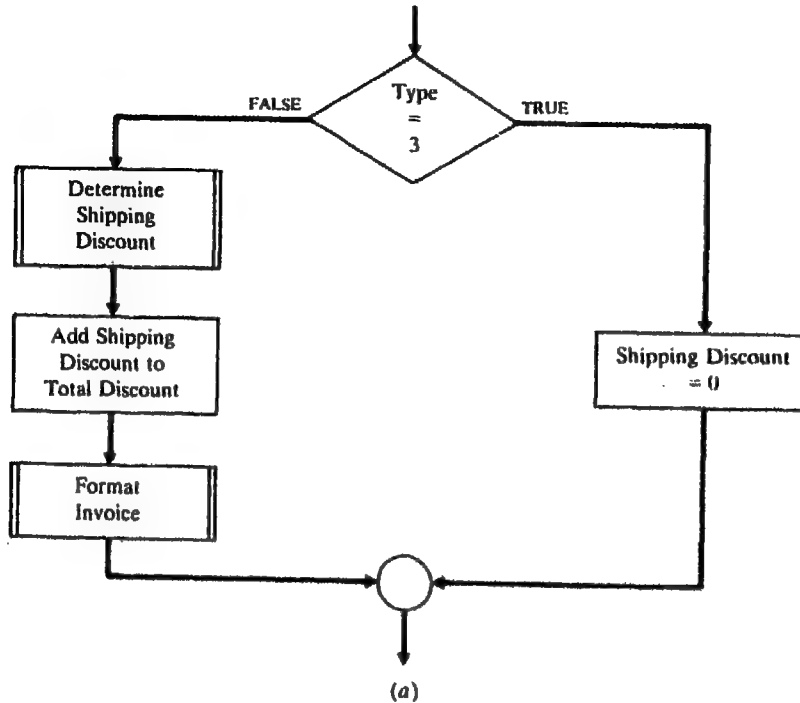
IF CUSTOMER-TYPE EQUAL "3"
  MOVE ZERO TO SHIPPING-DISCOUNT
ELSE
  PERFORM DETERMINE-SHIPPING-DISCOUNT

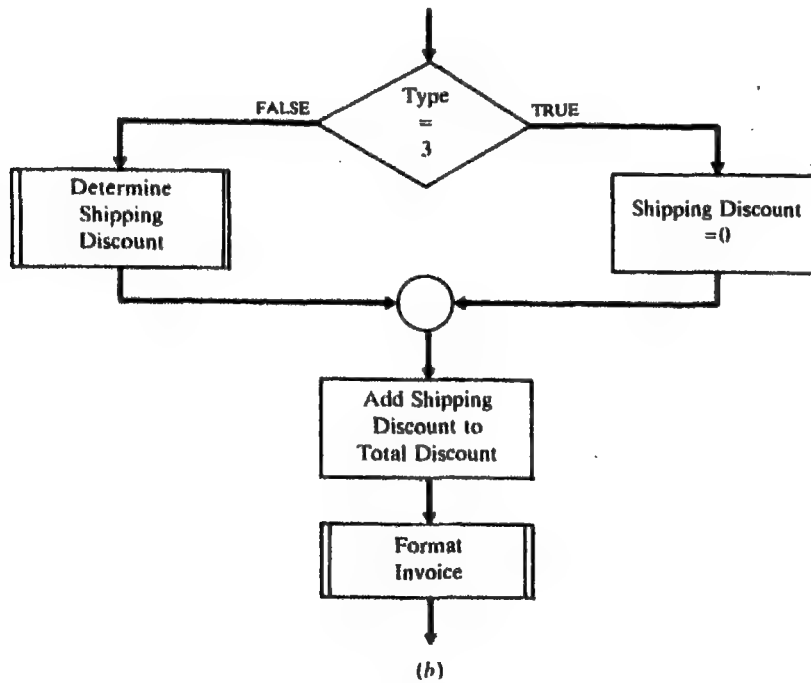
  ADD SHIPPING-DISCOUNT TO TOTAL-DISCOUNT
  PERFORM FORMAT-INVOICE

```

بدون النقطة .. تعتبر عبارة ADD وعبارة PERFORM جزءاً من ELSE (انظر ملحق جـ بالنسبة إلى كويل 1985 النمطي) .

٧ - ٥٤ ارسم خريطة مسار لجزء برنامج التمرين السابق (أ) بدون نقطة (ب) وبالنقطة . انظر شكل ٧ - ٢٢ .





شكل (٧ - ٢٢)

٧ - ٥٥ أعد كتابة عبارة IF التالية بدون استخدام NEXT SENTENCE .

```

IF NO-SPECIAL-DISCOUNTS
    NEXT SENTENCE
ELSE
    ADD 1 TO NUMBER-SPECIAL-CASES
    PERFORM DETERMINE-SPECIAL-DISCOUNT
IF NOT NO-SPECIAL-DISCOUNTS
    ADD 1 TO NUMBER-SPECIAL-CASES
    PERFORM DETERMINE-SPECIAL-DISCOUNT
    
```

وهناك حل أفضل ، وهو تعريف اسم شرطي جديد : SPECIAL - DISCOUNTS :

```

05 SPECIAL-DISCOUNT-SWITCH    PIC X.
   88 NO-SPECIAL-DISCOUNTS    VALUE "A".
   88 SPECIAL-DISCOUNTS       VALUE "B" THRU "G".
    
```

يمكن الآن حذف NOT كما يلي :

```

IF SPECIAL-DISCOUNTS
    ADD 1 TO NUMBER-SPECIAL-CASES
    PERFORM DETERMINE-SPECIAL-DISCOUNT
    
```


٧ - ٦ اكتب جزءاً من برنامج كويل لوضع YES في INVALID - RECORD - SW ، إذا كان أحد الحقول التالية غير عددي : ACCOUNT - NUMBER و TRANSACTION - CODE و AMOUNT .

```
IF ACCOUNT-NUMBER NOT NUMERIC
  OR TRANSACTION-CODE NOT NUMERIC
  OR AMOUNT NOT NUMERIC
  MOVE "YES" TO INVALID-RECORD-SW
```

٧ - ٧ صحح الشفرة التالية (ارجع للتمرين السابق) .

```
IF ACCOUNT-NUMBER NOT NUMERIC
  PERFORM INVALID-ACCOUNT
ELSE IF TRANSACTION-CODE NOT NUMERIC
  PERFORM INVALID-TRANSACTION
ELSE IF AMOUNT NOT NUMERIC
  PERFORM INVALID-AMOUNT
```

الخطأ هنا هو أن IF الخطية غير مناسبة لمثل هذا النوع من اختبار الصحة . بمجرد تحقق أحد الشروط .. يهمل بقية IF الخطية ، وهذا ليس مرغوباً فيه ؛ لأننا نريد الاستمرار في اختبار الصحة بعد اكتشاف أى خطأ بحيث يمكن معرفة أى أخطاء أخرى في السجل (وتصحيحها) .

والطريقة الصحيحة تستخدم مجموعة من عبارات IF المنفصلة على النحو التالي :

```
IF ACCOUNT-NUMBER NOT NUMERIC
  PERFORM INVALID-ACCOUNT

IF TRANSACTION-CODE NOT NUMERIC
  PERFORM INVALID-TRANSACTION

IF AMOUNT NOT NUMERIC
  PERFORM INVALID-AMOUNT
```

٧ - ٨ بمعرفة PICTURE والمحتويات التالية ، اذكر ما إذا كان عنصر البيانات عددياً أم حرفياً أم غير ذلك .

	PICTURE	Contents
(a)	S99	-82
(b)	XXX	"E T"
(c)	S99	82
(d)	99	82
(e)	99	+82
(f)	XXX	"C3B"
(g)	XXX	" "

(a) عددي (b) حرفي (c) لا هذا ولا ذاك (الصورة بها إشارة ، ويجب أن تكون بالمحتويات إشارة) . (d) عددي (e) لا هذا ولا ذاك (عندما لا تكون هناك إشارة في PICTURE فيجب ألا توجد إشارة في المحتويات) (f) لا هذا ولا ذاك (g) حرفي (الفراغات مسموح بها) .

٧ - ٥٩ اذكر ما اذا كان يجب التأكد من صحة عناصر البيانات التالية قبل التشغيل أم لا .

- (أ) قيمة مبلغ مودع بينك يقوم الصراف بإدخاله .
- (ب) الاسم البريدي المشترك في مجلة يتم إدخاله عن طريق إدخال البيانات .
- (ج) قيمة المشتريات على الحساب التي يدخلها موظف المحل البائع .
- (د) موازنة الرصيد الدائن للعميل من (ح) كما تقرأ من الملف الرئيسي للعملاء الموجود على قرص .
- (هـ) موازنة الحساب السابق للحساب الموجود في (أ) كما يقرأ من ملف العملاء الرئيسي ، الموجود على قرص .
- (و) كمية العنصر المطلوبة من قبل العميل ، ويتم إدخالها في مستند المصدر عن طريق إدخال البيانات .
- (ز) سعر العنصر في (و) الذي يقرأ من ملف مخزون رئيسي على قرص .
- (أ) الحقل يكون مهما ، ويتم إدخاله بواسطة مشغل الكمبيوتر . وعلى هذا يجب التأكد من صحة .
- (ب) هذا الحقل يدخله مشغل الكمبيوتر إلا أنه ليس مهما . يمكن أن تصل المجلة بصورة مناسبة إلى المقصد المقصود ، حتى إذا كان هناك خطأ في الاسم . كما أن اختبار الفئة لبيانات حرفية يمكن أن يكتشف رموزاً غير عددية فقط وليس خطأ في الهجاء . وعلى هذا .. فالتأكد من صحة هذا الحقل قد لا تستحق إجراؤها .
- (ج) يجب التأكد ستكون من صحته في أول مرة يتم تشغيله ، حيث إنه يمثل بيانات حرجية ويدخله مشغل كمبيوتر .
- (د) بافتراض أن كل البيانات موضوعة في الملف الرئيسي سبق التأكد من صحتها .. فإن إعادة التأكد من صحتها ستكون مضيعة لوقت الكمبيوتر .
- (هـ) نفس الشيء مثل (د) .
- (و) يجب أن يتم التأكد من صحته في أول مرة يجرى عليه تشغيل .
- (ز) نفس الشيء مثل (د) .

٧ - ٦٠ بين العلاقات الجبرية التالية كما تبدو عند كتابتها بعبارة IF من الكويل

$$(a) a + b \leq 10 \quad (b) 5a \geq \frac{b}{3} \quad (c) a \neq b \quad (d) \frac{a}{b} \leq c \quad (e) a \geq b$$

- (a) IF A + B NOT GREATER THAN 10
- (b) IF 5 * A NOT LESS THAN B / 3
- (c) IF A NOT EQUAL TO B
- (d) IF A / B NOT GREATER THAN C
- (e) IF A NOT LESS THAN B

٦١ - ٧ حدد إذا كان العنصر A أقل من ، أو يساوى ، أو أكبر من العنصر B .

	(a)	(b)	(c)	(d)	(e)	(f)	(g)
A	CAD	+72	HI	-003	-4	+34.5	+34.00
B	COB	0072	HIGH	-3	+4	+34	+34

(a) أقل من (يحفظ الترتيب الأبجدي)

(b) يساوى (مقارنة جبرية) .

(c) أقل من (HI = Hlbb)

(d) يساوى (مقارنة جبرية)

(e) أقل من (مقارنة جبرية)

(f) أكبر (مقارنة جبرية)

(g) يساوى (مقارنة جبرية)

٦٢ - ٧ أعد كتابة العبارات التالية : مستخدماً شروط إشارة بدلاً من شروط علاقات .

(a) IF AMOUNT-DUE IS GREATER THAN ZERO ... ;

(b) PERFORM POST-GENERAL- LEDGER UNTIL LEDGER-ACCOUNT IS EQUAL TO ZERO;

(c) IF QUANTITY-ON- HAND IS LESS THAN ZERO

(a) IF AMOUNT-DUE IS POSITIVE ... ; (b) PERFORM POST-GENERAL-LEDGER UNTIL LEDGER-ACCOUNT IS ZERO; (c) IF QUANTITY-ON-HAND NEGATIVE

لاحظ حذف IS من (c) .

٦٣ - ٧ أعد تعريفات البيانات التالية ، وعبارات IF و PERFORM ، مستخدماً أسماء شرطية بدلاً من شروط العلاقات :

(a) 05 CUSTOMER-ID PIC X(5).

IF CUSTOMER-ID NOT LESS THAN ZERO AND NOT GREATER THAN "1111"
PERFORM PROCESS-PREFERRED-CUSTOMER

(b) 05 TRANSACTION-CODE PIC X.

IF TRANSACTION-CODE EQUAL "1"
PERFORM PROCESS-DEPOSIT
ELSE IF TRANSACTION-CODE EQUAL "2"
PERFORM PROCESS-WITHDRAWAL

- (c) 05 REGION-CODE PIC X.
- IF REGION-CODE EQUAL "A" OR "B" OR "C"
 PERFORM SHIP-TO-NORTHEAST
 ELSE IF REGION-CODE GREATER THAN "C"
 AND LESS THAN "L"
 PERFORM SHIP-TO-SOUTHEAST
 ELSE IF REGION-CODE EQUAL "M" OR "P"
 PERFORM SHIP-WEST
- (d) 05 YEAR-IN-SCHOOL PIC 99.
- IF YEAR-IN-SCHOOL EQUAL 6
 PERFORM PROCESS-6TH-GRADER
 ELSE IF YEAR-IN-SCHOOL NOT LESS THAN 9 AND NOT
 GREATER THAN 12
 PERFORM PROCESS-HIGH-SCHOOL
- (e) 05 DEPRECIATION-TYPE PIC 9.
- IF DEPRECIATION-TYPE EQUAL 1 OR 2 OR 3
 PERFORM USE-STRAIGHT-LINE-VALUE
 ELSE IF DEPRECIATION-TYPE EQUAL 4
 PERFORM USE-DOUBLE-DECLINING-VALUE
- (a) 05 CUSTOMER-ID PIC X(5).
 88 PREFERRED-CUSTOMER VALUE ZERO THRU "1111".
- IF PREFERRED-CUSTOMER
 PERFORM PROCESS-PREFERRED-CUSTOMER
- (b) 05 TRANSACTION-CODE PIC X.
 88 DEPOSIT VALUE "1".
 88 WITHDRAWAL VALUE "2".
- IF DEPOSIT
 PERFORM PROCESS-DEPOSIT
 ELSE IF WITHDRAWAL
 PERFORM PROCESS-WITHDRAWAL
- (c) 05 REGION-CODE PIC X.
 88 NORTHEAST-REGION VALUE "A" THRU "C".
 88 SOUTHEAST-REGION VALUE "D" THRU "K".
 88 WESTERN-REGION VALUE "M" "P".
- IF NORTHEAST-REGION
 PERFORM SHIP-TO-NORTHEAST
 ELSE IF SOUTHEAST-REGION
 PERFORM SHIP-TO-SOUTHEAST
 ELSE IF WESTERN-REGION
 PERFORM SHIP-WEST
- (d) 05 YEAR-IN-SCHOOL PIC 99.
 88 6TH-GRADE VALUE 6.
 88 HIGH-SCHOOL VALUE 9 THRU 12.

```

IF 6TH-GRADE
  PERFORM PROCESS-6TH-GRADER
ELSE IF HIGH-SCHOOL
  PERFORM PROCESS-HIGH-SCHOOL.

```

- (e) 05 DEPRECIATION-TYPE PIC 9.
 88 STRAIGHT-LINE VALUE 1 THRU 3.
 88 DOUBLE-DECLINING VALUE 4.

```

IF STRAIGHT-LINE
  PERFORM USE-STRAIGHT-LINE-VALUE
ELSE IF DOUBLE-DECLINING
  PERFORM USE-DOUBLE-DECLINING-VALUE

```

٧ - ٦٤ صحح العبارات التالية ، والتي تستخدم الأسماء الشرطية من التمرين رقم ٦٣ استخداماً خاطئاً .

- (a) IF PREFERRED-CUSTOMER EQUAL "00111"
 (b) IF DEPOSIT
 MOVE SPACES TO WITHDRAWAL
 (c) IF HIGH-SCHOOL EQUAL 10

(أ) IF CUSTOMER-ID EQUAL "00111"
 لا يمكن استخدام اسم شرطي ، بدلا من اسم بيانات بأي حال من الأحوال في البرنامج . الاسم الشرطي هو اختصار فقط لشرط علاقة محدد .

(ب) IF DEPOSIT
 MOVE SPACES TO TRANSACTION-CODE

كما في (أ) .. يجب استخدام عنصر البيانات في عبارة MOVE ، وليس اسماً شرطياً .

(ج) IF YEAR-IN-SCHOOL EQUAL 10

لا يوجد اسم شرطي للعلاقة :

YEAR - IN - SCHOOL EQUAL 10 ، وعلى هذا .. يجب كتابة الشرط في عبارة IF . والبدل لذلك هو عمل شرط اسم شرطي كما يلي :

```

05 YEAR-IN-SCHOOL PIC 99.
88 10TH-GRADE VALUE 10.

```

```

IF 10TH-GRADE

```

٧ - ٦٥ هناك قاعدة في الكويل تنص على أن جزء VALUE لا يمكن استخدامه في قسم الملفات . هل ينطبق ذلك على الأسماء الشرطية ؟

لا ، تنطبق القاعدة على عناصر البيانات في قسم الملفات فقط . يمكن (ويجب) استخدام جزء VALUE في قسم الملفات عند تعريف أسماء شرطية (عناصر على المستوى 88) .

٧ - ٦٦ إذا كانت A, B, C أسماء شرطية .. افترض أن A, B صحيحان بينما C خطأ . حدد ما إذا كانت الشروط المركبة التالية صحيحة أم خطأ .

- | | |
|----------------------------|----------------------------|
| (a) A AND B OR A AND C | (b) A AND (B OR A) AND C |
| (c) A AND NOT C OR D | (d) A AND NOT (C OR D) |
| (e) A AND NOT C OR NOT D | (f) (A AND NOT C) OR NOT D |
| (g) A AND NOT (C OR NOT D) | (h) A AND C OR B AND D |
| (i) A AND B AND C | (j) A OR B OR C OR D |
| (k) A OR C AND B OR D | (l) (A OR C) AND (B OR D) |
| (m) A AND C OR B OR D | |

(a) في غياب الأقواس .. تنفذ AND أولاً (من اليسار إلى اليمين) يتبعها تنفيذ OR . وعلى هذا .. يتم تقويم التعبير كما يلي:
(A AND B) OR (A AND C) . (A AND B) صحيح وبالتالي فإن (A AND B) OR .. يكون صحيحاً كذلك .

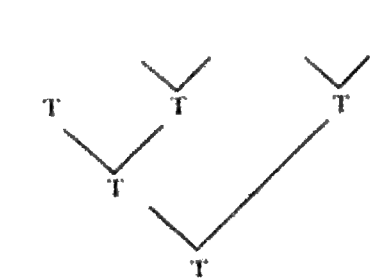
(b) يتم تقويم التعبير الموجود بين قوسين أولاً : (B OR A) صحيح . ثم ينفذ AND من اليسار لليمين : A AND (B OR A) يكون صحيحاً ، هذه النتيجة مرتبطة بواسطة AND مع C وينتج عنها نتيجة خطأ ، وهى النتيجة النهائية .

(c) يتم تقويم NOT أولاً لتعطي النتيجة صحيح ، وعلى ذلك تقويم A AND NOT ، وتكون صحيحة . D مع هذه النتيجة بواسطة OR ، وبالتالي فالنتيجة النهائية صحيحة .

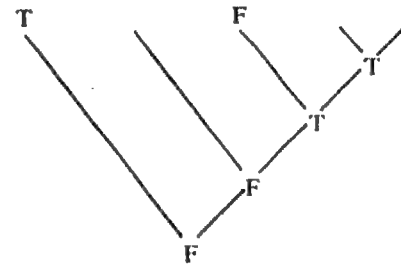
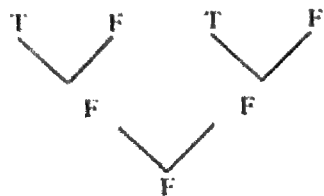
(d) (C OR D) تعطي نتيجة خطأ وبالتالي NOT (C OR D) تعطي نتيجة صحيحة . وهذه مرتبطة مع A بواسطة AND فتعطي النتيجة النهائية صحيحة .

(e) تنفذ NOT أولاً من اليسار لليمين نتيجة NOT C هي صحيح ، كما أن نتيجة NOT D صحيحة . ترتبط A مع NOT C باستخدام AND وتعطي النتيجة صحيحة . ترتبط هذه النتيجة بواسطة OR مع NOT D لتكون النتيجة النهائية صحيحة .

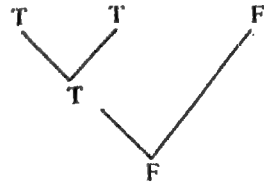
- (f) (A AND NOT C) OR NOT D (g) A AND NOT (C OR NOT D)



(h) A AND C OR B AND D



(i) A AND B AND C



لاحظ استخدام OR بدلا من AND : أحد الشروط البسيطة فقط يجب أن يكون صحيحا ليقع ITEM-A خارج المدى المحدد .

٧ - ٧٠ اكتب شرطا يكون صحيحا إذا ، وإذا فقط ، كان أى من - PAYMENT , DUE , PAYMENT , OLD - BALANCE غير عددي ،

IF OLD-BALANCE NOT NUMERIC
OR PAYMENT NOT NUMERIC
OR PAYMENT-DATE NOT NUMERIC

٧ - ٧١ اكتب شرطا يكون صحيحا إذا ، وإذا فقط ، كان A مساويا 1 ، و B مساويا 2 ، و D ليس مساويا 4 .

IF A EQUAL 1
AND B EQUAL 2
AND C EQUAL 3
AND D NOT EQUAL 4

٧ - ٧٢ اكتب شرطا يكون صحيحا إذا ، وإذا فقط ، لم يكن END-OF-FILE صحيحا ، وإما أن يكون BALANCE سالباً أو يكون PAYMENT مساويا صفر .

IF NOT END-OF-FILE
AND (BALANCE IS NEGATIVE OR PAYMENT IS ZERO)

الاقواس مطلوبة : حيث يمكن تقويم A AND B OR C على أنها (A AND B) OR C .

٧ - ٧٣ اكتب شرطا يكون صحيحا إذا ، وإذا فقط ، كانت YEAR تقع بين 85,82 ، وتقع MONTH بين 8,6 ، وتقع DAY بين 20,5 .

IF (YEAR NOT LESS THAN 82 AND NOT GREATER THAN 85)
AND (MONTH NOT LESS THAN 6 AND NOT GREATER THAN 8)
AND (DAY NOT LESS THAN 5 AND NOT GREATER THAN 20)

حيث إن المؤثر المنطقي AND هو المطلوب فقط فكل الاقواس اختيارية (بالرغم من أنها تحسن من التوضيح) .

٧٤ - ٧ بين كيف يمكن اختصار شروط العلاقات التالية :

- (a) A NOT EQUAL 1 AND A NOT EQUAL 2 AND A NOT EQUAL 3
- (b) A EQUAL 7 OR A EQUAL 8 AND B EQUAL 8
- (c) A EQUAL 1 OR A EQUAL 2 AND A EQUAL 3
- (d) A EQUAL 1 OR A EQUAL 2 OR A LESS THAN 9 OR B LESS THAN 10
- (e) A EQUAL 1 OR A LESS THAN 5 OR A GREATER THAN 8
- (f) A EQUAL B AND A LESS THAN C AND A LESS THAN D
- (a) A NOT EQUAL 1 AND 2 AND 3
- (b) A EQUAL 7 OR 8 AND B EQUAL 8
- (c) A EQUAL 1 OR 2 AND 3 (which can be logically reduced to A EQUAL 1)
- (d) A EQUAL 1 OR 2 OR LESS THAN 9 OR B LESS THAN 10
- (e) A EQUAL 1 OR LESS THAN 5 OR GREATER THAN 8
- (f) A EQUAL B AND LESS THAN C AND D

٧٥ - ٧ اكتب شروط العلاقات التالية في صوره غير مختصرة .

- (a) A EQUAL B OR C AND LESS THAN D
- (b) A GREATER THAN B OR D AND C OR LESS THAN B OR D
- (c) A LESS THAN B AND D AND C
- (d) A EQUAL B OR D AND C
- (e) A EQUAL B OR C AND LESS THAN D OR GREATER THAN E
- (f) A NOT EQUAL B OR D AND NOT LESS THAN C AND E
- (a) A EQUAL B OR A EQUAL C AND A LESS THAN D
- (b) A GREATER THAN B OR A GREATER THAN D AND A GREATER THAN C OR A LESS THAN B OR A LESS THAN D
- (c) A LESS THAN B AND A LESS THAN D AND A LESS THAN C
- (d) A EQUAL B OR A EQUAL D AND A EQUAL C
- (e) A EQUAL B OR A EQUAL C AND A LESS THAN D OR A GREATER THAN E
- (f) A NOT EQUAL B OR A NOT EQUAL D AND A NOT LESS THAN C AND A NOT LESS THAN E

٧٦ - ٧ راجع الترحيل المضلل التالي لتبين كيف تنفذ IF المتداخلة فعلا .

- (a) IF NOT END-OF-FILE
 - IF CREDIT-LINE-OK
 - PERFORM DETERMINE-CREDIT-LIMIT
 - PERFORM PRINT-TERMS
 - ELSE
 - PERFORM PRINT-FINAL-TOTALS

- (b) IF PAYMENT NOT NUMERIC
PERFORM INVALID-PAYMENT
IF DATE2 NOT NUMERIC
PERFORM INVALID-DATE2
ELSE
PERFORM APPLY-PAYMENT
IF BALANCE NEGATIVE
PERFORM GIVE-REFUND
ELSE
PERFORM FILE-BALANCE
- (c) IF A EQUAL B
MOVE 1 TO A
MOVE 2 TO B.
MOVE 3 TO C

IF B EQUAL C
MOVE 4 TO D
IF C EQUAL D
MOVE 5 TO E
ELSE
MOVE 6 TO F.
- (a) IF NOT END-OF-FILE
IF CREDIT-LINE-OK
PERFORM DETERMINE-CREDIT-LIMIT
PERFORM PRINT-TERMS
ELSE
PERFORM PRINT-FINAL-TOTALS
- (b) IF PAYMENT NOT NUMERIC
PERFORM INVALID-PAYMENT
IF DATE2 NOT NUMERIC
PERFORM INVALID-DATE2
ELSE
PERFORM APPLY-PAYMENT
IF BALANCE NEGATIVE
PERFORM GIVE-REFUND
ELSE
PERFORM FILE-BALANCE

هنا ، كما في (١) ، كل جزء ELSE متزامن مع أول IF تسبقه .

- (c) IF A EQUAL B
MOVE 1 TO A
MOVE 2 TO B

MOVE 3 TO C
IF B EQUAL C
MOVE 4 TO D
IF C EQUAL D
MOVE 5 TO E

```
ELSE
  MOVE 6 TO F
```

لاحظ أهمية النقطة في تغيير معنى IF المتداخلة . بدون النقطة بعد MOVE 2 TO B .. ينفذ التسلسل كما يلي :

```
IF A EQUAL B
  MOVE 1 TO A
  MOVE 2 TO B
  MOVE 3 TO C
  IF B EQUAL C
    MOVE 4 TO D
    IF C EQUAL D
      MOVE 5 TO E
    ELSE
      MOVE 6 TO F
```

٧ - ٧٧ اعد كتابة ما يلي دون استخدام NEXT SENTENCE ، صحح كذلك الترحيل :

```
IF A EQUAL B
  NEXT SENTENCE
ELSE
  PERFORM PARA-1
  PERFORM PARA-2.
  PERFORM PARA-3
```

```
IF A NOT EQUAL B
  PERFORM PARA-1
  PERFORM PARA-2
```

```
PERFORM PARA-3
```

٧ - ٧٨ أكتب شفرة هيكل الحالة التالي في مقطع مستخدما IF الخطية .

REGION-CODE Value	Action
1	PERFORM SHIP-NORTHEAST
2	PERFORM SHIP-SOUTHEAST
3	PERFORM SHIP-NORTHEAST
4	PERFORM SHIP-WEST

```
DETERMINE-SHIPPING.
  IF REGION-CODE EQUAL 1
    PERFORM SHIP-NORTHEAST
  ELSE IF REGION-CODE EQUAL 2
    PERFORM SHIP-SOUTHEAST
  ELSE IF REGION-CODE EQUAL 3
    PERFORM SHIP-NORTHEAST
  ELSE IF REGION-CODE EQUAL 4
    PERFORM SHIP-WEST
  ELSE
    PERFORM INVALID-CODE
```

أو بدمج الحالات :

```
DETERMINE-SHIPPING.
  IF REGION-CODE EQUAL 1 OR 3
    PERFORM SHIP-NORTHEAST
  ELSE IF...
```

٧ - ٧٩ وضع كيف ينفذ هيكل الحالة في التمرين السابق (٧ - ٧٨) .

PERFORM DETERMINE-SHIPPING

٧ - ٨٠ اعد كتابة ما يلي مستخدما ... UNTIL . PERFORM

```
PERFORM GET-INPUT
PERFORM GET-INPUT
PERFORM GET-INPUT
PERFORM GET-INPUT
PERFORM GET-INPUT
  VARYING TIMES-COUNTER
  FROM 1 BY 1
  UNTIL TIMES-COUNTER GREATER THAN 4
```

٧ - ٨١ كم عدد المرات التي ينفذ فيها GET-INPUT ?

```
PERFORM GET-INPUT
  VARYING TIMES-COUNTER FROM 1 BY 1
  UNTIL TIMES-COUNTER EQUAL 10
```

٩ مرات .

٧ - ٨٢ كم عدد المرات التي ينفذ فيها PARA-A ؟

```
MOVE 10 TO TIMES-COUNTER
PERFORM PARA-A TIMES-COUNTER TIMES
.....
PARA-A.
  MOVE 5 TO TIMES-COUNTER
  MOVE P TO Q
```

٧ - ٨٣ كم عدد المرات التي ينفذ فيها PARA - A ؟

PERFORM PARA-A
 VARYING TIMES-COUNTER FROM 1 BY 1
 UNTIL TIMES-COUNTER GREATER THAN 10

.....

PARA-A.
 MOVE P TO Q
 MOVE 11 TO TIMES-COUNTER

ينفذ PARA-A مرة واحدة . عندما تستخدم UNTIL .. VARYING .. PERFORM ، فأى تغيير فى عنصر بيانات VARYING ، أو فى قيمة BY أو فى قيمة UNTIL يؤثر على عدد مرات التكرار .
 ٧ - ٨٤ كم عدد المرات التى ينفذ فيها PARA-A ؟

MOVE 11 TO TIMES-COUNTER
 PERFORM PARA-A
 UNTIL TIMES-COUNTER GREATER THAN 10

لا ينفذ PARA-A على الإطلاق ، ويتم تقويم شرط UNTIL قبل أى PERFORM .
 ٧ - ٨٥ كم عدد مرات التى ينفذ فيها PARA-S ؟

001 MOVE ZERO TO TIMES-COUNTER
 PERFORM PARA-S
 UNTIL TIMES-COUNTER GREATER THAN 10

PARA-S.
 ADD 1 TO TIMES-COUNTER
 DISPLAY TIMES-COUNTER

١١ مرة : TIMES - COUNTER يعرض القيم 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 .
 ٧ - ٨٦ حدد قيم ITEM لكل تنفيذ للمقطع PARA-A :

- (a) PERFORM PARA-A VARYING ITEM FROM 1 BY 2
 UNTIL ITEM GREATER THAN 8
- (b) PERFORM PARA-A
 VARYING ITEM FROM 10 BY -3
 UNTIL ITEM LESS THAN -4
- (c) PERFORM PARA-A
 VARYING ITEM FROM 2.7 BY -.4
 UNTIL ITEM NEGATIVE
- (d) PERFORM PARA-A
 VARYING ITEM FROM 6 BY 2
 UNTIL ITEM EQUAL 11
- (e) PERFORM PARA-A
 VARYING ITEM FROM 5 BY 5
 UNTIL ITEM GREATER THAN 25

- (a) 7, 5, 3, 1 يتوقف التنفيذ عند $ITEM = 9$.
- (b) 2, 1, 4, 7, 10 يتوقف التنفيذ عند $ITEM = -5$.
- (c) 0.3, 0.7, 1.1, 1.5, 1.9, 2.3, 2.7 يتوقف التنفيذ عند $ITEM = -0.1$.
- (d) 16, 14, 12, 10, 8, 6 ,... حتى ينهى نظام التشغيل الوقت ، أو يزيد عدد السجلات عن العدد المحدد من قبل نظام التشغيل . هذه بكرة لا نهائية .
- (e) 50, 20, 15, 10, 5 يتوقف التنفيذ عند $ITEM = 30$.
- ٧ - ٨٧ حدد قيم B,A التي ينفذ عندها PARA .

PERFORM PARA
 VARYING A FROM 2 BY 2
 UNTIL A GREATER THAN 8
 AFTER B FROM 5.5 BY -0.5
 UNTIL B LESS THAN 4.0

A	2	2	2	2	4	4	4	4	6	6	6	6	8	8	8	8
B	5.5	5.0	4.5	4.0	5.5	5.0	4.5	4.0	5.5	5.0	4.5	4.0	5.5	5.0	4.5	4.0

ينتهي تنفيذ عبارة PERFORM عند $A=10, B=5.5$.

٧ - ٨٨ حدد قيم C,B,A التي ينفذ عندها PARA .

PERFORM PARA
 VARYING A FROM 2 BY 3
 UNTIL A GREATER THAN 7
 AFTER B FROM 5 BY -1
 UNTIL B LESS THAN 3
 AFTER C FROM 1 BY 1
 UNTIL C GREATER THAN 3

A	2	2	2	2	2	2	2	2	2	5	5	5	5	5	5	5	5
B	5	5	5	4	4	4	3	3	3	5	5	5	4	4	4	3	3
C	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2

ينتهي تنفيذ عبارة PERFORM عند $A=8, B=5, C=1$.

٧ - ٨٩ وضح الخطأ بالنسبة للأجزاء التالية :

- (a) PERFORM PARA-A
 VARYING ITEM-A FROM 1 BY 1 UNTIL ITEM-A GREATER THAN 10
 VARYING ITEM-B FROM 5 BY 2 UNTIL ITEM-B GREATER THAN 20

- (b) PERFORM PARA-A THRU PARA-C

PARA-C.

PARA-A.

(c) PERFORM PRODUCE-REPORT
UNTIL END-OF-FILE

```
PRODUCE-REPORT
  IF INPUT-FIELD NUMERIC
    PERFORM PROCESS-IT
  ELSE
    PERFORM INVALID-FIELD
```

INVALID-FIELD.
PERFORM WRITE-ERROR-MESSAGE
PERFORM READ-NEXT-RECORD
PERFORM PRODUCE-REPORT

(a) خطأ تكويني : PERFORM مع عناصر متغيرة عديدة يجب أن تكتب في صورة ... VARYING ... PERFORM . AFTER ... AFTER ...

(b) خطأ تكويني: يجب على مقطع (أو قسم) THRU أن يتبع واقعياً أول مقطع (أو قسم) في المدى .

(c) خطأ منطقي : **PERFORM** لها إعادة ذاتية . تنفذ عبارة **PERFORM** الأولية المقطع **PRODUCE - REPORT** طبقا لصحة المدخلات ، يمكن للمقطع **PRODUCE - REPORT** أن يتفذ **PERFORM** المقطع **INVALID - FIELD** . عند ذلك .. يتفذ المقطع **INVALID - FIELD** المقطع **PRODUCE - REPORT** . وعلى هذا .. فعن الممكن تنفيذ **PERFORM** في **INVALID - FIELD** قبل انتهاء أول تنفيذ لها . تنفيذ عبارة **PERFORM** ، التي لا تكون قد انتهت تنفيذها السابق (لان التحكم لم يعد من المقطع المنفذ) عادة ما تنتج عن دورة نهائية .

٧ - ٩٠ أكتب شفرة شبكية لبرنامج موازنة دفتر شيكات . يدخل الخوارزمي الموازنة القيمة تتبعها مجموعة من رموز العمليات الجارية (D للإيداع ، و C لقيمة الشيك المسحوب) والمبالغ . حتى نهاية الملف .. المخرجات هي إجمالى الإيداعات وإجمالى الشيكات والموازنة الجديدة .

```

read old-balance
set total-deposits, total-checks to zero
set end-sw to "no"
read transaction-code, amount
  at end set end-sw to "yes"
perform until end-sw = "yes"
  if transaction-code = "D"
    add amount to total-deposits
  else
    if transaction-code = "C"
      add amount to total-checks
    else
      print "invalid transaction code"
    endif
  endif
endperform
new-balance = old-balance + total-deposits - total-checks
print old-balance, total-deposits, total-checks, new-balance
stop

```

٧ - ٩١ اكتب شفرة شبيهة لتنفيذ جدول القرارات الموجود فى شكل ٧ - ١٨

```

if account not active
    perform disapprove-purchase
else
    if last month's payment received
        if purchase not greater than credit limit
            perform approve-purchase
        else
            perform approve-purchase
            perform refer-to-credit-office
        endif
    else
        if purchase not greater than credit limit
            perform approve-purchase
            perform refer-to-credit-office
        else
            perform disapprove-purchase
        endif
    endif
endif
endif

```

بينما يكون هذا الحل مباشرا ، إلا أن هناك حلاً أكثر إيجازاً ، وهو ما يلى :

```

if account not active
    perform disapprove-purchase
else
    if last payment received and purchase within credit limit
        perform approve-purchase
    else
        if last payment not received and purchase not within credit limit
            perform disapprove-purchase
        else
            perform approve-purchase
            perform refer-to-credit-office
        endif
    endif
endif
endif

```

٧ - ٩٢ بمعرفة الشروط التالية والإجراءات .. ارسم جدول قرارات لقبول عمل فى تشغيل البيانات .

<u>الشروط</u>	<u>الإجراءات</u>
(١) الراتب والمزايا مقبولة	(١) اقبل العمل
(٢) تعليم وتدريب أكثر متاح	(٢) أرفض العمل
(٣) فرص تقدم جيدة	(٣) ناقش للوصول الى مزايا أفضل

انظر شكل ٧ - ٢٣ لرؤية حل ممكن .

Salary	T	T	T	T	F	F	F	F
Education	T	T	F	F	T	T	F	F
Advancement	T	F	T	F	T	F	T	F
Accept	x	x	x	x	x			
Refuse						x	x	x
Negotiate				x	x	x	x	

شكل (٧ - ٢٣)

مشاكل متكاملة

٧ - ٩٣ اكتب برنامجا يمكن استخدامه في أحد البنوك في طباعة جداول دفع حصص ديون العملاء . مداخل البرنامج هي قيمة أصل الدين ، ومعدل الفائدة السنوية (كنسبة مئوية) والقسط الشهري المرغوب في دفعه . يطبع البرنامج تقريراً يسرد المعلومات التالية لكل شهر لفترة إعادة دفع الدين :

(١) رقم القسط .

(٢) الموازنة القديمة (قبل دفع هذا القسط) .

(٣) قيمة القسط .

(٤) قيمة ما دفع من أصل الدين [(٣) - (٥)] .

$$(٥) \text{ قيمة ما دفع من الفوائد } [(٢) \times \frac{\text{معدل الفائدة}}{12}]$$

(٦) الموازنة الجديدة [(٤) - (٥)] . يجب أن يطبع البرنامج 20 سطراً ، بينها مسافات مزدوجة في الصفحة الواحدة . استخدم عناوين أعمدة مناسبة وأرقام صفحات ، وتاريخ التنفيذ وما إلى ذلك من معلومات . انظر شكل ٧ - ٢٤ للحل ، وشكل (٧-٢٥) لعينة مخرجات .

```

00001 IDENTIFICATION DIVISION.
00002
00003 PROGRAM-ID. MORTGAGE.
00004
00005 AUTHOR. LARRY NEWCOMER.
00006 INSTALLATION. PENN STATE UNIVERSITY -- YORK CAMPUS.
00007 DATE-WRITTEN. MAY 1983.
00008 SECURITY. NONE.
00009
00010 ENVIRONMENT DIVISION.
00011
00012 CONFIGURATION SECTION.
00013 SOURCE-COMPUTER. IBM-3081.
00014 OBJECT-COMPUTER. IBM-3081.
00015
00016 INPUT-OUTPUT SECTION.
00017 FILE-CONTROL.
```

```

00018
00019      SELECT TERMS-FILE
00020          ASSIGN TO TERMS
00021          ORGANIZATION IS SEQUENTIAL
00022          ACCESS IS SEQUENTIAL
00023
00024      SELECT PAYMENT-REPORT
00025          ASSIGN TO PAYMENTS
00026          ORGANIZATION IS SEQUENTIAL
00027          ACCESS IS SEQUENTIAL
00028
00029
00030      DATA DIVISION.
00031
00032      FILE SECTION.
00033
00034      FD  TERMS-FILE
00035          RECORD CONTAINS 80 CHARACTERS
00036          LABEL RECORDS ARE OMITTED
00037
00038
00039      01  TERMS-RECORD.
00040          05  TERMS-PRINCIPAL          PIC S9(7)V99.
00041          05  TERMS-INTEREST-RATE     PIC SV999.
00042          05  TERMS-PAYMENT          PIC S9(5)V99.
00043          05  FILLER                  PIC X(61).
00044
00045      FD  PAYMENT-REPORT
00046          RECORD CONTAINS 132 CHARACTERS
00047          LABEL RECORDS ARE OMITTED
00048          LINAGE IS 60
00049              WITH FOOTING AT 59
00050              LINES AT TOP 3
00051              LINES AT BOTTOM 3
00052
00053
00054      01  PAYMENT-RECORD                PIC X(132).
00055
00056      WORKING-STORAGE SECTION.
00057
00058      01  WS-SWITCHES-AND-COUNTERS.
00059          05  WS-PAGE-NUMBER            PIC S999          COMP-3.
00060          05  WS-TERMS-FILE-END-SW     PIC XXX.
00061              88  EMPTY-TERMS-FILE    VALUE "YES".
00062          05  WS-FINAL-PAYMENT-SW      PIC XXX.
00063              88  FINAL-PAYMENT       VALUE "YES".
00064              88  REGULAR-PAYMENT     VALUE "NO".
00065
00066      01  WS-COMPUTATION-WORK-AREAS.
00067          05  WS-INTEREST-AMOUNT       PIC S9(5)V99      COMP-3.
00068          05  WS-PAYMENT-AMOUNT        PIC S9(5)V99      COMP-3.
00069          05  WS-BALANCE               PIC S9(7)V99      COMP-3.
00070          05  WS-PAYMENT-NUMBER        PIC S9(3)         COMP-3.
00071          05  WS-INTEREST-RATE        PIC SV999         COMP-3.
00072          05  WS-PRINCIPAL-AMOUNT     PIC S9(5)V99      COMP-3.
00073
00074      01  WS-DATE.
00075          05  SYSTEM-YY               PIC 99.
00076          05  SYSTEM-MM               PIC 99.
00077          05  SYSTEM-DD               PIC 99.
00078
00079      01  WS-HEADING-LINE-1.
00080          05  FILLER                   PIC X(10)         VALUE SPACES.
00081          05  FILLER                   PIC X(17)         VALUE "MORTGAGE PAYMENTS".
00082
00083          05  FILLER                   PIC X(1)         VALUE SPACES.
00084          05  WS-HEADING-MM            PIC Z9.

```

```

00085      05 FILLER                      PIC X          VALUE "/".
00086      05 WS-HEADING-DD                PIC 99.
00087      05 FILLER                      PIC X          VALUE "/".
00088      05 WS-HEADING-YY                PIC 99.
00089      05 FILLER                      PIC X(4)        VALUE SPACES.
00090      05 FILLER                      PIC X(5)        VALUE "PAGE ".
00091      05 WS-HEADING-PAGE-NO            PIC ZZ9.
00092      05 FILLER                      PIC X(84)       VALUE SPACES.
00093
00094      01 WS-HEADING-LINE-2.
00095      05 FILLER                      PIC X(10)
00096      VALUE "PAYMENT #".
00097      05 FILLER                      PIC X(2)        VALUE SPACES.
00098      05 FILLER                      PIC X(15)
00099      VALUE "OLD BALANCE".
00100      05 FILLER                      PIC X(2)        VALUE SPACES.
00101      05 FILLER                      PIC X(10)
00102      VALUE "PAYMENT".
00103      05 FILLER                      PIC X(2)        VALUE SPACES.
00104      05 FILLER                      PIC X(10)
00105      VALUE "PRINCIPAL".
00106      05 FILLER                      PIC X(2)        VALUE SPACES.
00107      05 FILLER                      PIC X(10)
00108      VALUE "INTEREST".
00109      05 FILLER                      PIC X(2)        VALUE SPACES.
00110      05 FILLER                      PIC X(12)
00111      VALUE "NEW BALANCE".
00112      05 FILLER                      PIC X(55)       VALUE SPACES.
00113
00114      01 WS-DETAIL-LINE.
00115      05 FILLER                      PIC X(3)        VALUE SPACES.
00116      05 WS-DETAIL-NUMBER             PIC Z9.
00117      05 FILLER                      PIC X(7)        VALUE SPACES.
00118      05 WS-DETAIL-OLD-BAL            PIC Z,ZZZ,ZZZ.99.
00119      05 FILLER                      PIC X(5)        VALUE SPACES.
00120      05 WS-DETAIL-PAYMENT            PIC ZZ,ZZZ.99.
00121      05 FILLER                      PIC X(3)        VALUE SPACES.
00122      05 WS-DETAIL-PRINCIPAL         PIC ZZ,ZZZ.99.
00123      05 FILLER                      PIC X(3)        VALUE SPACES.
00124      05 WS-DETAIL-INTEREST          PIC ZZ,ZZZ.99.
00125      05 FILLER                      PIC X(3)        VALUE SPACES.
00126      05 WS-DETAIL-NEW-BAL          PIC Z,ZZZ,ZZZ.99.
00127      05 FILLER                      PIC X(55)       VALUE SPACES.
00128
00129      PROCEDURE DIVISION.
00130
00131      000-EXECUTIVE-MODULE.
00132
00133      PERFORM 100-GET-TERMS
00134      PERFORM 200-PRODUCE-DETAIL-LINE
00135      UNTIL FINAL-PAYMENT
00136      PERFORM 300-PRODUCE-FINAL-PAYMENT
00137      PERFORM 400-TERMINATE
00138      STOP RUN
00139
00140
00141      100-GET-TERMS.
00142
00143      MOVE "NO" TO WS-TERMS-FILE-END-SW
00144      WS-FINAL-PAYMENT-SW
00145      MOVE ZERO TO WS-PAGE-NUMBER
00146      WS-PAYMENT-NUMBER
00147      OPEN INPUT TERMS-FILE
00148      OUTPUT PAYMENT-REPORT
00149      READ TERMS-FILE

```

```

00150             AT END
00151             MOVE "YES" TO WS-TERMS-FILE-END-SW
00152
00153             IF EMPTY-TERMS-FILE
00154                 MOVE "YES" TO WS-FINAL-PAYMENT-SW
00155                 DISPLAY "*** EMPTY TERMS FILE ***"
00156             ELSE
00157                 MOVE TERMS-PRINCIPAL      TO WS-BALANCE
00158                 MOVE TERMS-INTEREST-RATE TO WS-INTEREST-RATE
00159                 MOVE TERMS-PAYMENT       TO WS-PAYMENT-AMOUNT
00160                 ACCEPT WS-DATE FROM DATE
00161                 MOVE SYSTEM-YY           TO WS-HEADING-YY
00162                 MOVE SYSTEM-MM           TO WS-HEADING-MM
00163                 MOVE SYSTEM-DD           TO WS-HEADING-DD
00164                 PERFORM 600-PRODUCE-HEADINGS
00165
00166
00167             200-PRODUCE-DETAIL-LINE.
00168
00169                 ADD 1 TO WS-PAYMENT-NUMBER
00170                 COMPUTE WS-INTEREST-AMOUNT =
00171                     (WS-BALANCE * WS-INTEREST-RATE) / 12
00172                 SUBTRACT WS-INTEREST-AMOUNT FROM WS-PAYMENT-AMOUNT
00173                 GIVING WS-PRINCIPAL-AMOUNT
00174                 IF WS-PRINCIPAL-AMOUNT NOT LESS THAN WS-BALANCE
00175                     MOVE "YES" TO WS-FINAL-PAYMENT-SW
00176
00177                 IF REGULAR-PAYMENT
00178                     MOVE WS-PAYMENT-NUMBER TO WS-DETAIL-NUMBER
00179                     MOVE WS-BALANCE        TO WS-DETAIL-OLD-BAL
00180                     MOVE WS-PAYMENT-AMOUNT TO WS-DETAIL-PAYMENT
00181                     MOVE WS-PRINCIPAL-AMOUNT TO WS-DETAIL-PRINCIPAL
00182                     MOVE WS-INTEREST-AMOUNT TO WS-DETAIL-INTEREST
00183                     SUBTRACT WS-PRINCIPAL-AMOUNT FROM WS-BALANCE
00184                     GIVING WS-DETAIL-NEW-BAL
00185                     WS-BALANCE
00186                     PERFORM 500-PRINT-LINE
00187
00188
00189             300-PRODUCE-FINAL-PAYMENT.
00190
00191                 MOVE WS-PAYMENT-NUMBER TO WS-DETAIL-NUMBER
00192                 MOVE WS-BALANCE        TO WS-DETAIL-OLD-BAL
00193                                     WS-DETAIL-PRINCIPAL
00194                 ADD WS-BALANCE WS-INTEREST-AMOUNT
00195                 GIVING WS-DETAIL-PAYMENT
00196                 MOVE WS-INTEREST-AMOUNT TO WS-DETAIL-INTEREST
00197                 MOVE ZERO                TO WS-DETAIL-NEW-BAL
00198                 PERFORM 500-PRINT-LINE
00199
00200
00201             PERFORM 100-GET-TERMS
00202             PERFORM 200-PRODUCE-DETAIL-LINE
00203             UNTIL FINAL-PAYMENT
00204             PERFORM 300-PRODUCE-FINAL-PAYMENT
00205             PERFORM 400-TERMINATE
00206             STOP RUN
00207
00208
00209             100-GET-TERMS.
00210
00211                 MOVE "NO" TO WS-TERMS-FILE-END-SW
00212                 WS-FINAL-PAYMENT-SW
00213

```

```

00144
00145      MOVE ZERO TO WS-PAGE-NUMBER
00146      WS-PAYMENT-NUMBER
00147      OPEN      INPUT  TERMS-FILE
00148      OUTPUT  PAYMENT-REPORT
00149      READ TERMS-FILE
00150      AT END
00151      MOVE "YES" TO WS-TERMS-FILE-END-SW
00152
00153      IF EMPTY-TERMS-FILE
00154      MOVE "YES" TO WS-FINAL-PAYMENT-SW
00155      DISPLAY "*** EMPTY TERMS FILE ***"
00156      ELSE
00157      MOVE TERMS-PRINCIPAL      TO WS-BALANCE
00158      MOVE TERMS-INTEREST-RATE TO WS-INTEREST-RATE
00159      MOVE TERMS-PAYMENT      TO WS-PAYMENT-AMOUNT
00160      ACCEPT WS-DATE FROM DATE
00161      MOVE SYSTEM-YY          TO WS-HEADING-YY
00162      MOVE SYSTEM-MM          TO WS-HEADING-MM
00163      MOVE SYSTEM-DD          TO WS-HEADING-DD
00164      PERFORM 600-PRODUCE-HEADINGS
00165
00166
00167      200-PRODUCE-DETAIL-LINE.
00168
00169      ADD 1 TO WS-PAYMENT-NUMBER
00170      COMPUTE WS-INTEREST-AMOUNT =
00171      (WS-BALANCE * WS-INTEREST-RATE) / 12
00172      SUBTRACT WS-INTEREST-AMOUNT FROM WS-PAYMENT-AMOUNT
00173      GIVING WS-PRINCIPAL-AMOUNT
00174      IF WS-PRINCIPAL-AMOUNT NOT LESS THAN WS-BALANCE
00175      MOVE "YES" TO WS-FINAL-PAYMENT-SW
00176
00177      IF REGULAR-PAYMENT
00178      MOVE WS-PAYMENT-NUMBER      TO WS-DETAIL-NUMBER
00179      MOVE WS-BALANCE             TO WS-DETAIL-OLD-BAL
00180      MOVE WS-PAYMENT-AMOUNT     TO WS-DETAIL-PAYMENT
00181      MOVE WS-PRINCIPAL-AMOUNT  TO WS-DETAIL-PRINCIPAL
00182      MOVE WS-INTEREST-AMOUNT   TO WS-DETAIL-INTEREST
00183      SUBTRACT WS-PRINCIPAL-AMOUNT FROM WS-BALANCE
00184      GIVING WS-DETAIL-NEW-BAL
00185      WS-BALANCE
00186      PERFORM 500-PRINT-LINE
00187
00188
00189      300-PRODUCE-FINAL-PAYMENT.
00190
00191      MOVE WS-PAYMENT-NUMBER      TO WS-DETAIL-NUMBER
00192      MOVE WS-BALANCE             TO WS-DETAIL-OLD-BAL
00193      WS-DETAIL-PRINCIPAL
00194      ADD WS-BALANCE WS-INTEREST-AMOUNT
00195      GIVING WS-DETAIL-PAYMENT
00196      MOVE WS-INTEREST-AMOUNT     TO WS-DETAIL-INTEREST
00197      MOVE ZERO                   TO WS-DETAIL-NEW-BAL
00198      PERFORM 500-PRINT-LINE
00199
00200
00201      400-TERMINATE.
00202
00203      CLOSE  TERMS-FILE
00204      PAYMENT-REPORT
00205
00206

```

```

00207      500-PRINT-LINE.
00208
00209      WRITE PAYMENT-RECORD
00210      FROM WS-DETAIL-LINE
00211      AFTER ADVANCING 2 LINES
00212      AT END-OF-PAGE
00213      PERFORM 600-PRODUCE-HEADINGS
00214
00215
00216      600-PRODUCE-HEADINGS.
00217
00218      ADD 1 TO WS-PAGE-NUMBER
00219      MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE-NO
00220      WRITE PAYMENT-RECORD
00221      FROM WS-HEADING-LINE-1
00222      AFTER ADVANCING PAGE
00223      WRITE PAYMENT-RECORD
00224      FROM WS-HEADING-LINE-2
00225      AFTER ADVANCING 2 LINES
00226

```

شكل (٧ - ٢٤)

MORTGAGE PAYMENTS		5/09/83	PAGE 1		
PAYMENT #	OLD BALANCE	PAYMENT	PRINCIPAL	INTEREST	NEW BALANCE
1	6,000.00	1,000.00	950.00	50.00	5,050.00
2	5,050.00	1,000.00	957.92	42.08	4,092.08
3	4,092.08	1,000.00	965.90	34.10	3,126.18
4	3,126.18	1,000.00	973.95	26.05	2,152.23
5	2,152.23	1,000.00	982.07	17.93	1,170.16
6	1,170.16	1,000.00	990.25	9.75	179.91
7	179.91	181.40	179.91	1.49	.00

شكل (٧ - ٢٥)

نماذج برهجة

٧ - ٩٤ بمعرفة سجلات البيع التالية الموجودة على ملف بطاقات .

Field	Data type & length
sales ID	X(5)
closing date	X(6) in the form yymmdd
amount this	S9(4)V99
year	
amount last	S9(4)V99
year	
unused	57 bytes

اطبع تقريراً يحتوى على عناوين مناسبة ، و سطرين فقط :

(١) تعريف المبيعات ، والكمية هذا العام ، والكمية العام الماضى ، والزيادة عن العام الماضى ، كل هذا للبائع الذى لديه أقصى زيادة فى المبيعات .

(٢) نفس المعلومات لكنها للبائع الذى حقق أقل زيادة فى المبيعات .

الزيادة فى المبيعات معرفة على أنها = الكمية هذا العام ، مطروحا منها كمية العام الماضى . من الممكن فى (1) ، ومن المحتمل فى (2) أن تكون الزيادة فى المبيعات سالبة ، أى أن هناك قلة فى المبيعات .

٧ - ٩٥ عدل التمرين رقم ٩٤ بحيث يطبع البرنامج إجمالى مبيعات هذا العام ، وإجمالى مبيعات العام الماضى ، ومتوسط مبيعات هذا العام ، ومتوسط مبيعات العام الماضى . اجعل مخرجاتك واقعية وقابلة للاستخدام بقدر الإمكان .

٧ - ٩٦ عدل التمرين رقم ٩٣ بحيث يطبع البرنامج ، بعد كل الشهر الثانى عشر ، إجمالى ما دفع فى الأصل، وإجمالى ما دفع فى الفائدة أثناء فترة الاثنى عشر شهراً السابقة . كما يجب أن يطبع كذلك إجمالى الفائدة الموزعة على القرض فى نهاية التقرير.

٧ - ٩٧ اكتب برنامجاً للتأكد من صحة (التنقيح) ، ملف مدخلات التمرين رقم ٩٤ يجب أن يطبع البرنامج تقريراً ، يسرد سجلات الملف التى تحتوى على حقول غير عديدة فقط . يجب وضع خط تحت الحقول غير الصحيحة فى التقرير . تذكر إنه يمكن أن يكون هناك حقل واحد ، أو 2 أو 3 ، أو الأربعة .. كلها غير صحيح فى نفس السجل . اطلع عناوين مناسبة للتقرير ومعها تاريخ التنفيذ ، ورقم الصفحة وعناوين الأعمدة . اختبر برنامجك بدقة لكل مجموعات الخليط الممكنة لسجلات المدخلات .

الفصل الثامن

إعداد البرنامج :

منهج الأجزاء ، ومن القمة إلى القاعدة

Program Development: The Top-Down, Modular Approach

حتى الآن ترك القارئ لوسائله الخاصة بكيفية مواجهة مشكلة البرمجة . يقدم هذا الفصل عملية تطوير برامج فعالة ، سبق اختبارها ، والتي يمكن أن تفيد المبتدئين والمحترفين كذلك .

٨ - ١ الخطوة الأولى : تعريف المشكلة

يجب على المبرمج أن يكتسب ويتعلم كل مواد التصميم التي أنتجها محللو النظم الذين تعاملوا مع المشكلة . مثل هذه المواد تشمل خرائط تخطيطات السجلات ، وخرائط مسافات الطابع ، والسرد الوصفي بالإنجليزية للبرامج . ويجب أن يعد المبرمج (أو المحلل) مثل هذه المواد بنفسه .

٨ - ٢ الخطوة الثانية : تصميم عام للبرنامج

الخطوة التالية هي إنتاج تصميم أجزاء من أعلى لأسفل (من القمة إلى القاعدة) للبرنامج وتوثيقه بخريطة هيكل - Structure Chart (تسمى كذلك جيولاً مرئياً للمحتويات) Visual table of contents (VTOC) أو خريطة هرمية (hierarchy chart) . ويسمى التصميم بأنه تصميم أجزاء modular لأنه يجرىء البرنامج إلى قطع ، والأجزاء modules ، يؤدي كل منها وظيفة معرفة تعريف جيداً ، ويمكن كتابة كل جزء كمقطع كويل . ويسمى التصميم من القمة إلى القاعدة top-down ؛ لأنه يركز على وظائف المستوى الأعلى أولاً (الشاملة أو المتحكم) التي يجب أن يؤديها البرنامج ، ويعامل وظائف المستوى الأدنى (المحددة التفصيلية) فيما بعد .

مثال ٨ - ١ :

الصورة النهائية لخريطة الهيكل لبرنامج تقرير مبيعات مبينه في شكل ٨ - ١ . إذا كانت كتابة كل جزء (كل مستطيل) كمقطع كويل .. فتحدد ، الخطوط الواصلة .. أى المقاطع تنفذ أى مقاطع أخرى . لاحظ أن كل جزء يحمل اسماً وصفيًا يعرف

وظيفة ورقم الجزء ورقم تعريف الخليط المستخدم شرطه لرقم الجزء، واسمه يجب ألا يزيد طوله عن 30 رمزا بالنسبة لاسم مقطع الكويل . لاحظ الركن المظلل الذي يعرف جزءاً (مقطوعاً) يسمى مُنْقَذَ بواسطة أكثر من جزء واحد من أجزاء الخريطة مثل 230 ، الذي يستدعى بواسطة كل من 100 و 220 . بالطبع يظهر PRINT - HEADINGS - 230 مرة واحدة فقط في برنامج مصدر الكويل ، ونستخدم أجزاء مكررة في خريطة الهيكل لحفظ الخريطة في شكل هرمى (شجرى) .

ماذا جرى في عقل المبرمج الذي أعد شكل ٨ - ١ جزءاً جزءاً ، من القمة إلى القاعدة ؟

١ - المشكلة الحالية هي تقرير مبيعات طبقاً للمنطقة ، وعلى هذا .. فإننى أضع جزءاً في قمة خريطة الهيكل ، وأضع له الرقم 000 ، واسميه Produce Sales by Region .

000

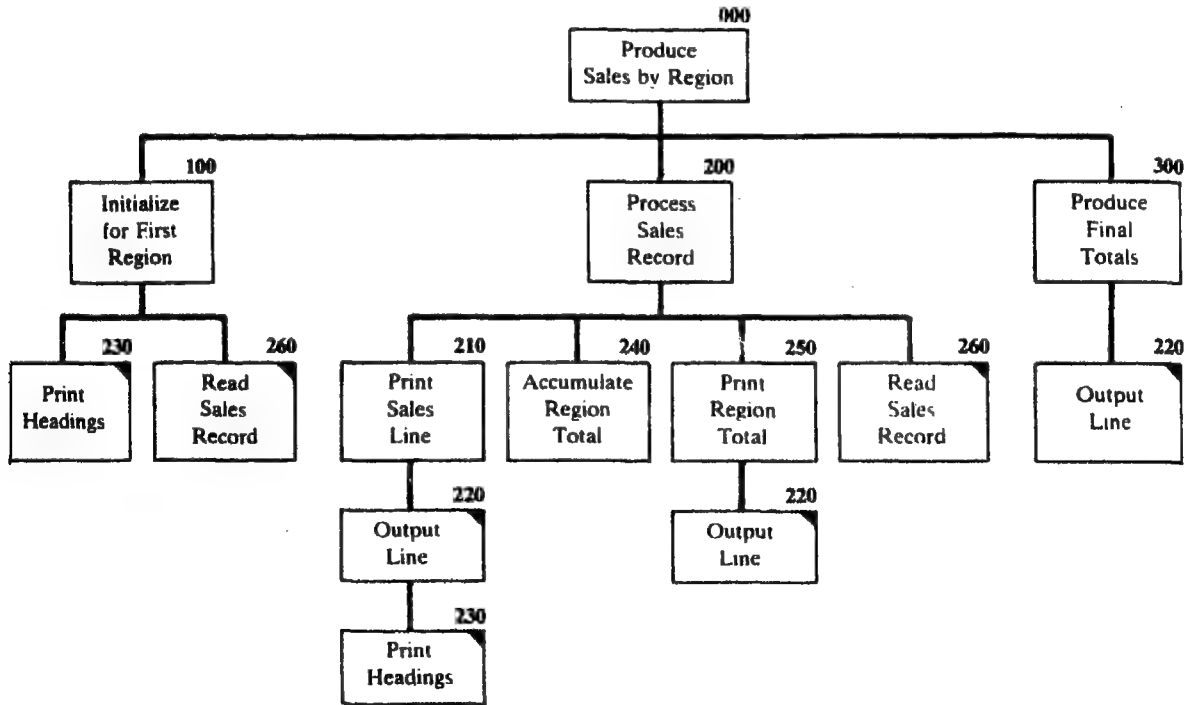
Produce Sales by Region

٢ - والآن .. ما الوظائف التى يجب أن تؤدي لى يتم إنتاج تقرير مبيعات طبقاً للمنطقة ؟ يبدو لى أن هناك ثلاث خطوات رئيسية : إعداد البرنامج لتشغيل بيانات لأول منطقة ، وتشغيل سجلات مبيعات فردية ، ومعاملة الإجماليات النهائية عند نهاية العمل . (لا توجد طريقة صحيحة وأخرى خطأ لتجزئة المشكلة إلى أجزاء فردية ، وبالرغم من أن بعض الحلول أفضل من البعض الآخر ، فلا يوجد بصفة عامة حل واحد أفضل) . بعد تعريف الوظائف الثلاثة الرئيسية التى تؤدي بواسطة PRODUCE - SALES - BY - REGION - 000 ، وفصل هذه الوظائف فى أجزاء خاصة بها ، تشبه خريطة الهيكل شكل ٨ - ٢ .

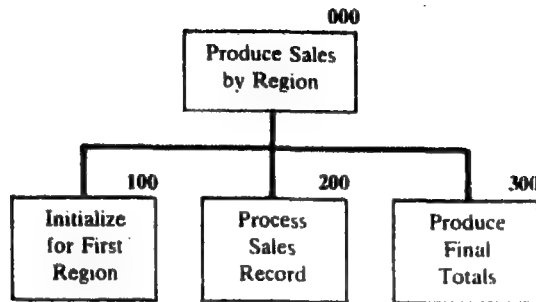
٣ - بتجزئة الجزء على المستوى 0 إلى وظائف تحكم منطقية رئيسية للبرنامج ، فإننى أكرر نفس الشيء على كل جزء على المستوى 1 . أى الوظائف المعرفة مستقلة ، والتى يجب أن تؤدي لبدء تشغيل أول منطقة ؟ حسناً .. على البرنامج أن يطبع عناوين لبدء التقرير ، ويجب كذلك أن يدخل أول سجل مبيعات ليذكر أى المناطق هي الأولى . وبالمثل .. لتشغيل سجل مبيعات ، يجب على البرنامج أن (١) يطبع سطر مبيعات فى التقرير ، سارداً محتويات سجل المبيعات (٢) هذا ويرسم إجمالى المنطقة بإضافة كمية المبيعات من سجل المبيعات هذا (٣) ويطبع ، إذا كانت هناك حاجة لذلك ، إجمالى المنطقة (إذا ما تغيرت المنطقة) .

٤ - ويدخل سجل المبيعات التالى . وعلى هذا .. يصبح عندى أجزاء على المستوى 2 (الأجزاء 230 و 260 و 210 و 240 و 250 و 260 و 220) . أقوم الآن بتجزئة هذه الأجزاء على المستوى ٢ الى وظائف جزئية لإنتاج أجزاء أخرى على المستوى 30 ، وهكذا حتى لا تحتوى الأجزاء على جزئية تافهة .

تخدم خريطة الهيكل الكاملة كنموذج لتنظيم جزء الإجراءات للبرنامج . وقد أوضحت الخبرة أن محاولة توفير الوقت بترك هذه المرحلة من التصميم - أو محاولة كتابة الشفرة أولاً ثم عمل التصميم ثانياً - تنتهى دائماً وبكل تأكيد بوقت أطول ، وتكلفة أكثر فى برامج بها أخطاء وغير متكافئة .



شكل (٨ - ١)



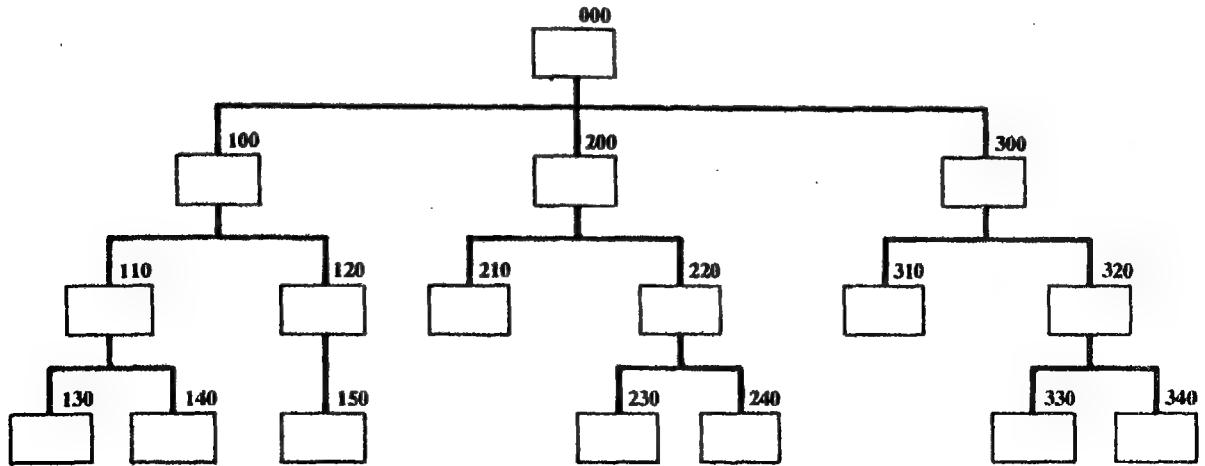
شكل (٨ - ٢)

خطوط إرشادية عند عمل خرائط الهيكل

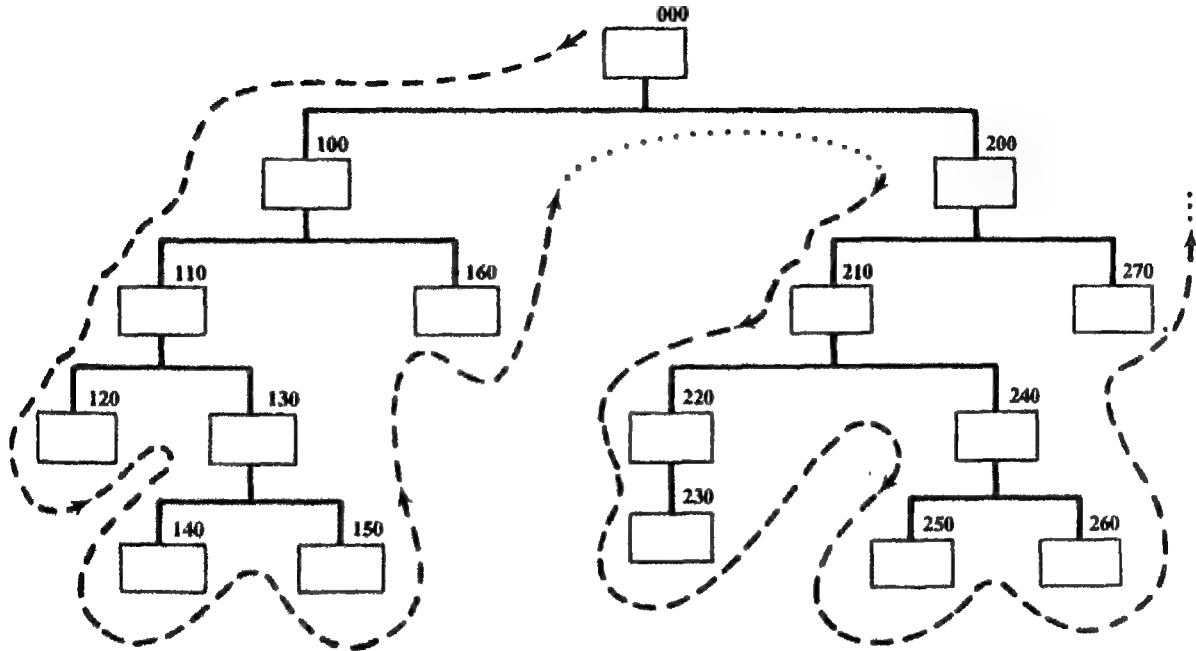
- ١ - وظيفة واحدة للجزء الواحد .
- ٢ - ضع رقماً لكل جزء . يقدم مثال ٨ - ٢ ومثال ٨ - ٣ طريقتين لوضع الأرقام . إذا ظهر أحد الأجزاء أكثر من مرة واحدة

فى خريطة الهيكل (أى إنه إذا استدعى بواسطة أكثر من جزء واحد) ، حده فى كل مكان بالرقم الذى يصاحب حدوثه الأكثر منطقية أو الأكثر تكرارا . كذلك يجب ان تحدد الأجزاء المكررة بصفة خاصة (كما فى شكل ٨ - ١) .

مثال ٨ - ٢ : فى الترقيم الأفقى horizontal numbering (شكل ٨ - ٣) وضعت رقم 000 للمستوى 0 . وضعت الأرقام 100, 200, 300 , ... للمستوى 1 . سلالات 100 ترقم مستوى بعد مستوى 110, 120, 130, وكذلك سلالات 200 ترقم مستوى بعد مستوى 210, 220, 230, الخ . لاحظ أن الترقيم يمكن تحقيقه من القمة إلى القاعدة ، مع تطوير الهيكل .



شكل (٨ - ٣)



شكل (٨ - ٤)

مثال ٨ - ٣ :

فى الترقيم الرأسى vertical numbering (شكل ٨ - ٤) لا يتقدم الشخص مستوى بمستوى ، لكنه يدور اليسار (كما يبدو فى الخريطة) عند كل وصلة junction .

وعلى هذا .. يرقم المستوى 0 بالرقم 000 والدوران اليسار ، لمقابلة الأجزاء 100 و 110 و 120 ... وفى نهاية السطر تحدث عودة إلى آخر وصلة آخر أول ما ترك دون أخذه . استمر فى الترقيم بزيادة 10 حتى تستهلك كل سلاطة 100 . عند ذلك ابدأ الرقم 200 (الاول من ناحية اليسار 000 لم يؤخذ حتى الآن) ، استخدم نفس القاعدة فى ترقيم سلاطات 200 - وهكذا . والطريقة السهلة لعمل ذلك هى رسم منحنى مستمر على حافة محتويات الخريطة (أنظر شكل ٨ - ٤) ، عند ذلك تستطيع عمل الترقيم عبر هذا المنحنى .

الترقيم الرأسى 2 الذى يميل إلى حفظ الأجزاء المنادية ، والأجزاء المنادى عليها قريبة من بعضها البعض ، يوصى باستخدامه فى نظم التخزين الافتراضى virtual storage (افحص دليل المورد المتاح لك) .

٣ - اعمل أجزاء متماسكة وظيفيا functionally cohesive ، والتي تعنى أن كل عبارة فى الجزء تسهم بطريقة مباشرة فى أداء وظيفة فردية للجزء .

٤ - اعمل أجزاء منخفضة فى التقارن Coupling ، وبمعنى آخر .. فى كلمات أخرى ، حاول أن تقلل عدد عناصر البيانات التى يتم الاتصال بها بمقطعين كويل أو أكثر .

٥ - يجب أن يتطلب الجزء أكثر من 50 سطرا من شفرة الكويل (أى صفحة واحدة من قائمة برنامج المصدر) .
يجب أن يتم تقويم أى جزء أكبر من صفحة واحدة لمعرفة ما إذا كان يجب نقل بعض عملياته إلى أجزاء على مستوى أقل تستدعى بواسطة هذا الجزء أم لا .

٦ - يجب أن تستدعى الاجزاء أجزاء أقل منها فى المستوى .

٧ - جزء التنفيذ executive module الموجود على المستوى 0 يكون مسئولاً بصفة عامة عن تكرار استدعاء نموذج على المستوى 1) ، وهذا يعد دورة البرنامج الرئيسية . وعادة ما تكون هذه الدورة من نوع « نفذ حتى نهاية الملف » .

٨ - حاول أن تستخدم جزء READ واحداً فقط للمدخلات ، وجزء WRITE واحداً فقط للمخرجات ، يمكن استدعاء جزء READ أو جزء WRITE المناسب بواسطة أى جزء ، عندما تكون هناك حاجة الى عملية مدخلات أو عملية مخرجات . هناك استثناء يحدث فى حالة ملفات الطباعة ، والتي يمكن لاحد الاجزاء أن يكتب WRITE عناوين ، وجزء آخر يكتب سطوراً تفصيلية .

٩ - إذا ما أصبح أحد الأجزاء التى لا تتكرر تافها (مثل وجود قلة من العبارات فيه) اعتبر حذفه ونقل عباراته إلى جزء آخر يستدعيه .

٨ - ٣ الخطوة الثالثة : تصميم تفصيلى للبرنامج

بعد إنتاج نموذج عام للبرنامج فى صورة خريطة هيكل (VTOC) تصبح الخطوة التالية فى إعداد البرنامج ، هى تصميم المنطق التفصيلى لكل جزء . يمكن عمل ذلك باستخدام خرائط مسار مرتبة أو شفرة شبيهة ، كما سبق ذكره فى الفصل السابع ، إلا أنه توجد وسيلة أكثر فائدة ، وهى خريطة هرمية المدخلات والتشغيل والمخرجات hierarchical input - process - output chart ، والتي تعرف بخريطة هيبو HIPO . وتدمج صفحة فردية من خريطة هيبو (شكل ٨ - ٥) وصف الشفرة الشبيهة للمنطق التفصيلى لأحد الأجزاء ، مع تحديد المدخلات أو عناصر بيانات مخزن العمل التى يحتاجها الجزء ، وتحديد المخرجات أو عناصر بيانات مخزن العمل التى ينتجها أو يغيرها الجزء .

يجب ملاحظة أن أحد عناصر البيانات يمكن أن يظهر في كل من مجموعة المدخلات ومجموعة المخرجات (إذا كان مستخرجاً بواسطة الجزء كما أن الجزء ، يجرى تغييراً عليه) . تعد صفحات هييو المختلفة من VTOC في صورة من القمة إلى القاعدة . انظر شكل ٨ - ١١ لرؤية خريطة هييو فعلية .
عند اتمام خريطة هييو .. يمكن اختبار الشفرة الشبيهة الموجودة في كل مجموعة مكتبا لتحديد وتصحيح ما يمكن من الأخطاء قبل بدء كتابة واختبار الشفرة

٨ - ٤ الخطوة الرابعة : إعداد خطة لكتابة الشفرة وإجراء الاختبارات

يمكن ، ويجب ، أن يطبق منهج من القمة إلى القاعدة في كتابة واختبار البرنامج كما حدث في تصميمه . وهذا يعني ما يلي:

- ١ - كتابة شفرة الأجزاء مرتفعة المستوى أولا ، بعد اختبار هذه الأجزاء وتصحيحها ، يمكن كتابة الأجزاء منخفضة المستوى واختبارها .
- ٢ - يجب ألا يكتب الجزء إلا إذا كتبت الأجزاء المحصورة بينه وبين جزء التنفيذ (المستوى ٠٠٠) واختبرت كذلك .
- ٣ - الجزء الذي ينادى على أجزاء أخرى لم تكتب بعد يمكن اختياره باستخدام أجزاء برنامج وهمية program stubs ، تحاكي تنفيذ الجزء المنادى عليه .

عادة ما يحتوى مثل هذا الجزء الذى يحاكي جزءاً من البرنامج على اسم المقطع تليه عبارة DISPLAY فقط تطبع رسالة تبين أن المقطع قد نفذ . وفى بعض الأحيان .. تقوم مثل هذه الأجزاء بوضع حالات المفاتيح اللازمة للجزء المنادى . (انظر شكل ٨ - ١٢ ، الأسطر من رقم ١٤٥ حتى ١٥٨ ، وشكل ٨ - ١٣ ، الأسطر من رقم ١٩٣ وحتى ٢١٥) .

يفضل بعض المبرمجين كتابة شفرة الأجزاء واختبارها بمستوى (أى فى تسلسل الترقيم الأفقى ، انظر مثال ٨-٢) . يستخدم البعض الآخر ترتيب الترقيم الرأسى (انظر مثال ٨ - ٣) . وطالما أن القاعدة رقم ٢ سالفة الذكر متبعة .. فإى خليط من هذين المنهجين يكون جيداً .

شكل (٨ - ٥)

Designer _____	System/Program Name _____	Date _____
Module No. _____	Module Name _____	

Input	Process	Output

الفصل الثامن : إعداد البرنامج : منهج الأجزاء ومن القمة إلى القاعدة ٢٠١

مثال ٨ - ٤ :

فيما يلي خطة كتابة شفرة برنامج شكل ٨ - ١ واختباره .

رقم التنفيذ	الأجزاء	الاختبار ب ...
1	000	كل الملفات فارغة مع استخدام أجزاء تحاكي 100 و 200 و 300.
2	100, 200, 300	بيانات منطقة مبيعات واحدة ، واستخدام أجزاء تحاكي الأجزاء 220, 250, 240, 210, 260, 230
3	210, 220, 230, 260, 240, 250	بيانات لمنطقتين .
4		(أ) كل الملفات فارغة .
		(ب) بيانات كل المناطق - بما في ذلك منطقة واحدة بها سجل واحد فقط .
		(ج) بيانات كافية لتسبب في عمل سريان زائد للصفحة

٨ - ٥ الخطوة الخامسة : تجميع بيانات الاختبارات

في بعض الأحيان .. يعد المبرمج بيانات الاختبارات ، وفي بعض الأحيان الأخرى .. يمكن إنتاج هذه البيانات بواسطة برنامج خاص ، ويمكن في بعض الأحيان الحصول عليها بنسخها من ملفات حقيقة تستخدمها برامج موجودة فعلا . عندما تجمع بيانات الاختبارات .. يجب ان يحدد المبرمج المخرجات الصحيحة المناظرة لهذه البيانات ، بحيث يمكن تعريف أي تعارضات بينها وبين المخرجات الفعلية .. ليتم تصحيح البرنامج .

وفي الحالة المثالية ، عند نهاية عملية الاختبارات .. يجب أن يتم اختبار كل مسار منطقي خلال كل جزء بصفة خاصة ، ويجب أن تقدم بيانات الاختبارات التي تأخذ كل IF خلال جزء true ، بينما تستدعي بيانات أخرى جزء false . يختبر كل تكرار UNTIL ... PERFORM لعدد مرات التكرار صفر ، 1 ، 2 ... إلخ (بإجمالي عدد مرات تكراره) . قد يكون من المستحيل للبرامج الكبيرة تقديم بيانات كافية لاختبار كل المسارات المنطقية الممكنة في وقت مناسب . في هذه الحالة ، يجب اختبار المسارات الأكثر أهمية ، والأكثر استخداما على الأقل بدقة .

٨ - ٦ الخطوة السادسة : كتابة الشفرة وعمل الاختبارات من القمة إلى القاعدة

يجب أن يكون البرنامج مكتوبا الآن ومختبرا ، جزءا بجزء ، طبقا للتسلسل الذي أعد في خطة كتابة الشفرة وإجراء الاختبارات .. وباستخدام بيانات الاختبارات التي أجرى عليها تحليل . انظر المسألة رقم 30 من هذا الفصل لرؤية تنفيذ هذه الخطوة .

٨ - ٧ الخطوة السابعة : إكمال توثيق البرنامج

بعد اختبار البرنامج وتصحيحه تماما ، يجب أن تجمع خريطة الهيكل وخريطة هيبو ، وخطة كتابة الشفرة وإجراء الاختبارات والبيانات الاختبارية وتنفيذه الاختبارات مع المخرجات وقوائم برنامج المصدر . ويجب إدخال أي تغيير داخل على البرنامج أثناء عملية الاختبار والتصحيح في VTOC و HIPO .

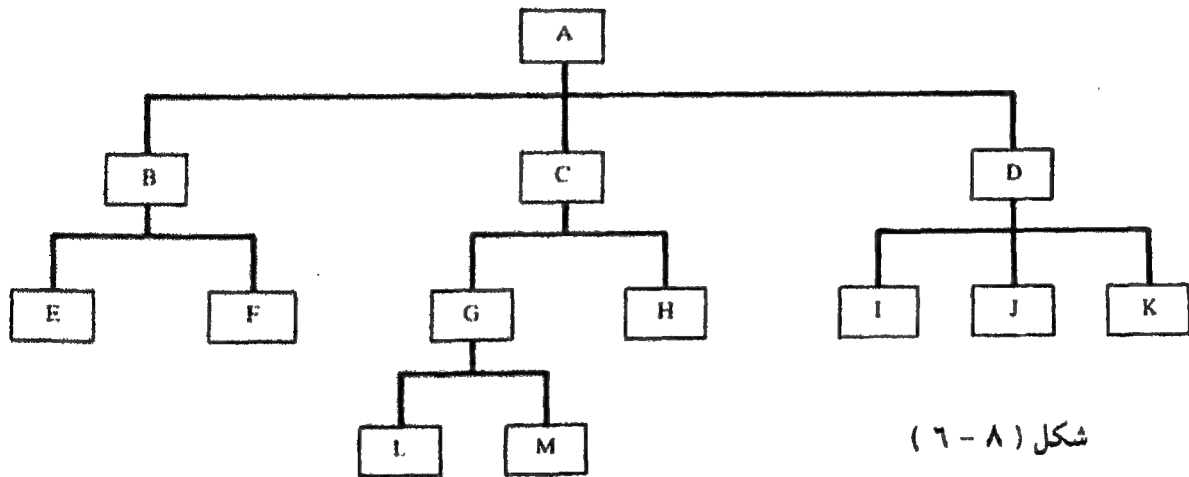
يجب حفظ كل الوثائق ليستخدمها مبرمجو الصيانة المحددين لإجراء تعديلات على البرنامج فيما بعد .

اسئلة مراجعة

- ٨ - ١ وضع أول خطوة في إعداد برنامج فردي .
- ٨ - ٢ ما خريطة الهيكل (VTOC) ؟ كيف تستخدم في التصميم العام للبرنامج ؟
- ٨ - ٣ ما معنى التصميم من أعلى لأسفل ، أو من القمة إلى القاعدة ؟
- ٨ - ٤ ما هو الجزء module ؟
- ٨ - ٥ عرف مفهوم المستوى في خريطة الهيكل .
- ٨ - ٦ قارن مع المضاهاة بين طريقة الترقيم الأفقي ، وطريقة الترقيم الرأسى :
- ٨ - ٧ وضع أهمية القاعدة « جزء واحد لوظيفة واحدة »
- ٨ - ٨ لماذا يجب أن تعرض الأجزاء تماسكا مرتفعا ؟
- ٨ - ٩ ما هو جزء التنفيذ ؟
- ٨ - ١٠ ماذا يوصى بوجود عبارة READ واحدة، أو عبارة WRITE واحدة لل ملف المستخدم في البرنامج ؟ كيف يمكن تحقيق ذلك ؟
- ٨ - ١١ وضع استخدام خريطة HIPO في التصميم التفصيلي للبرنامج .
- ٨ - ١٢ ناقش ما يحدث في مجموعات المدخلات والمخرجات من خريطة هيرو .
- ٨ - ١٣ ماذا تعنى كتابة الشفرة وعمل الاختبارات من القمة الى القاعدة ؟
- ٨ - ١٤ ما جزء البرنامج الذى يحاكي جزء ا لم تكتب شفرته ؟
- ٨ - ١٥ قارن المنهج الأفقى لكتابة الشفرة وعمل الاختبارات من القمة إلى القاعدة مع المنهج الرأسى .
- ٨ - ١٦ ما هدف اختبار البرنامج ؟ وضع كيف يجب أن تتغير بيانات الاختبارات أثناء عملية الاختبار من القمة إلى القاعدة .
- ٨ - ١٧ ما أهمية توثيق البرنامج ؟ وضع كيف يمكن لوسائل تصميم البرنامج أن تعمل فيما بعد كتوثيق للبرنامج .

مسائل محلولة

- ٨ - ١٨ ضع أرقاماً لأجزاء شكل ٨ - ٦ (أ) أفقياً (ب) ورأسياً .



شكل (٨ - ٦)

SAMPLE-PARA.

```
MOVE SPACES TO SALARIED-MESSAGE-AREA
MOVE ZEROS TO TOTAL-SALARY-AMOUNT
PERFORM READ-TIME-CARD
ADD TIME-CARD-HOURS TO TOTAL-HOURS
IF TIME-CARD-ID NOT NUMERIC
    MOVE "YES" TO INVALID-SWITCH
```

```
PERFORM COMPUTE-TAX
IF LINE-COUNT-ON-REPORT GREATER THAN 30
    PERFORM PRINT-HEADINGS
```

الجزء التماسك هو الجزء الذى تسهم كل عبارة فيه فى أداء وظيفة معرفة جيداً . الجزء الذى (١) يضع قيماً ابتدائية لبعض عناصر البيانات (٢) ويدخل سجلاً (٣) ويرسم إحصائيات (٤) ويتحقق من صحة حقل (٥) ويحسب ضريبة الدخل (٦) ويعامل عناوين الطباعة عند الوصول إلى نهاية الصفحة ، يكون بالتأكيد ذا تماسك منخفض للغاية .

٨ - ٢٠ ماهى الوظائف التى يؤديها جزء التنفيذ فى أحد تطبيقات الأعمال التقليدية ؟

عادة ما يؤدي جزء التنفيذ أربع وظائف .

SAMPLE-EXECUTIVE-MODULE.

```
PERFORM INITIALIZATION-MODULE
PERFORM MAIN-PROCESSING-MODULE
    UNTIL END-OF-FILE
PERFORM TERMINATION-MODULE
STOP RUN
```

يجعل الجزء INITIALIZATION - MODULE البرنامج يبدأ (تفتح الملفات ، توضع أصفار فى العدادات ، وفى مناطق الإحصائيات ، يقرأ أول سجل منطقي .. إلخ) . الجزء TERMINATION - MODULE يحضر البرنامج إلى تعليق مرتب (طباعة إحصائيات نهائية ونهايات التقرير ، وإغلاق الملفات ... إلخ) . الجزء MAIN - PROCESSING - MODULE يؤدي الغرض الرئيسى من البرنامج (مثل طباعة شيكات ، والتأكد من صحة ملف المدخلات ، وتجديد الملف الرئيسى وطباعة التقرير) . إذا شملت عملية الإعداد initialization والإنهاء termination عملاً بسيطاً نسبياً ، فلا تحتاج أن تنفذ كأجزاء مستقلة ويمكن كتابتها داخل جزء التنفيذ نفسه .

```

100-INITIALIZE-PROGRAM
  MOVE ZERO TO...
  MOVE SPACES TO...
  OPEN...
  READ INPUT-FILE
    AT END MOVE "YES" TO END-FILE-SW

  IF END-FILE-SW NOT EQUAL "YES"
    MOVE A TO B

  .....

210-PRODUCE-REPORT-LINE.
  MOVE...
  ADD...
  READ INPUT-FILE
    AT END MOVE "YES" TO END-FILE-SW

```

هنا استخدمت عبارة READ منفصلة كلما كان مطلوباً إدخال سجل منطقي . ويمكن أن يكون هذا غير مقنع إذا تغيرت مواصفات المدخلات والمخرجات للبرنامج ؛ حيث يجب على مبرمج الصيانة أن يتصيد - خلال محتويات البرنامج - كل عبارة READ . هناك طريقة أفضل ، وهي وضع عبارة READ للملف في مقطع خاص بها :

```

100-INITIALIZE-PROGRAM.
  MOVE ZERO TO...
  MOVE SPACES TO...
  OPEN...
  PERFORM GET-INPUT-RECORD
  IF END-FILE-SW NOT EQUAL "YES"
    MOVE A TO B

  .....

210-PRODUCE-REPORT-LINE.
  MOVE...
  ADD...
  PERFORM GET-INPUT-RECORD

  .....

GET-INPUT-RECORD.
  READ INPUT-FILE
    AT END
      MOVE "YES" TO END-FILE-SW

```

الفصل الثامن : إعداد البرنامج : منهج الأجزاء ومن القمة إلى القاعدة ٣.٥

٨ - ٢٢ أحد أجزاء شكل ٨ - ١ تافها بالتأكيد ، عرفه .

يحتوى ACCUMULATE - REGION - TOTAL - 240 على عبارة ADD واحدة ، يمكن إدخالها ببساطة في الجزء المنادى ، 200 - PROCESS - SALES - RECORD

٨ - ٢٣ لماذا يجب أن تبدأ بيانات الاختبارات بسيطة ؟

البرنامج الجديد به عديد من الأخطاء بصفة عامة . وفي البداية تتسبب بيانات الاختبارات المعقدة في تداخل الأخطاء مع بعضها البعض بطرق ، يمكن أن تكون مضللة وتجعل الأخطاء صعبة التعريف . أكثر من ذلك في عملية الاختبار ، عندما تصحح معظم الأخطاء .. فإنه يمكن استخدام بيانات اختبارات أكثر تعقيدا .

٨ - ٢٤ افرض أن C, B, A هي حقول في سجل مدخلات .

حدد مجموعة من قيم اختبارية لكل من C, B, A تختبر كل المسارات المنطقية في الشفرة التالية :

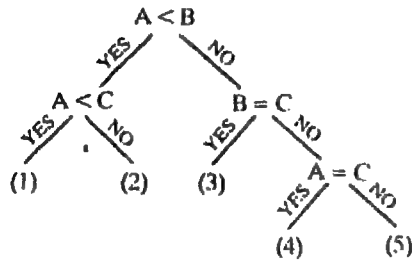
```

IF A LESS THAN B
  MOVE ...
  ADD ...
  IF A LESS THAN C
    PERFORM ...
  ELSE
    PERFORM ...
ELSE
  SUBTRACT ...
  IF B EQUAL C
    MOVE ...
  ELSE
    PERFORM ...
  IF A EQUAL C
    MOVE ...

```

ارسم شجرة قرارات decision tree لعبارات IF المتداخلة ، شكل ٨ - ٧ ، المسارات المنطقية معطاة بالمسارات من الجذر A < B إلى الأوراق أو نقاط النهاية للشجرة . وتبدو هناك 5 مسارات ، موضع بها بيانات اختيارية في جدول ٨ - ١

جدول (٨ - ١)



شكل (٨ - ٧)

Path	A	B	C
(1)	1	2	3
(2)	1	2	0
(3)	2	1	1
(4)	2	1	2
(5)	2	1	3

٨ - ٢٥ تصل عديد من برامج إعداد تقارير الأعمال إلى معلوماتها عن طريق تشغيل ملفات على شرائط أو أقراص . تبني هذه البرامج حول دورة رئيسية ، يتم إدخال سجل منطقي فيها وتشغيله ، وإذا كان مناسباً .. يتم إخراج النتائج . هناك منهجان أساسيان لتصميم المنطق لمثل هذا البرنامج . في المنهج الأول .. يدخل جسم الدورة الرئيسية (١) سجلاً منطقياً (٢) يختبر نهاية الملف (٣) إذا لم تحدث نهاية الملف .. يقوم بتشغيل السجل المنطقي الذي تم إدخاله على التو . أكتب شفرة شبيهة لتوضيح هذا المنطق للبرنامج الذي يسرد الرواتب الشهرية للعاملين بعد قراءته الملف الرئيسي للعاملين .

```
set end-of-file-switch "off"
open employee-master, report-file
perform until end-of-file-switch is "on" (main loop):
  read employee-master-record
  at end: set end-of-file-switch "on"
  if end-of-file-switch is "off"
    perform process-employee-record
    perform output-employee-results
  endif
endperform
close employee-master, report-file
stop
```

اختبار نهاية الملف حرجة في هذا المنهج حيث إنها تتم قبل تشغيل السجل المنطقي (انظر مثال ٦ - ١٠) . في بعض النظم .. لاتزال منطقة السجل المنطقي (FD) تحتوى آخر سجل تمت قراءته عندما حدثت نهاية الملف . وفي هذه الحالة يمكن أن يؤدي الفشل في اختبار نهاية الملف قبل تشغيل سجل إلى تشغيل آخر سجل مرتين.

٨ - ٢٦ المنهج الأساسي الثاني لنفس نوع البرامج التي سبق مناقشتها في المسألة السابقة يستخدم قراءة أولية priming read، وهي قراءة أولية للملف المنفذ قبل بدء دورة البرنامج الرئيسية . وحيث إنه تم إدخال سجل منطقي فعلاً عندما تنفذ الدورة الرئيسية .. فإن أول شيء تؤديه الدورة الرئيسية هو (١) تشغيل سجل منطقي وإخراج النتائج ، عند ذلك (٢) تقرأ السجل المنطقي التالي (الذي يجرى عليه تشغيل عندما تتكرر الدورة عند الخطوة الأولى) . المنهج القراءة الأولية ميزتان :

(١) عدم وجود الحاجة إلى اختبار نهاية الملف داخل الدورة .

(٢) حيث إن أول سجل تم إدخاله منفصلاً .. فمن الممكن أن يعالج معالجة خاصة (انظر المسألة التالية) . أعد كتابة المسألة السابقة مع استخدام القراءة الأولية .

```
set end-of-file-switch "off"
open employee-master, report-file
read employee-master record (priming read)
  at end: set end-of-file-switch "on"
... (any special processing required for the first
  logical record could be done here)
perform until end-of-file-switch is "on" (main loop):
  perform process-employee-record
  perform output-employee-results
  read employee-master record
  at end: set end-of-file-switch "on"
endperform
close employee-master, report-file
stop
```

٢٠٧ الفصل الثامن : إعداد البرنامج : منهج الأجزاء ومن القمة إلى القاعدة

٨ - ٢٧ بين كيف يمكن استخدام منهج المسألة رقم ٢٥ من هذا الفصل (الذى لا يستخدم قراءة أولية) إذا كان أول سجل منطقى فى الملف يتطلب تشغيلًا خاصًا .

```
set end-of-file-switch to "off"
set first-time-switch to "on"
open employee-master, report-file
perform until end-of-file-switch is "on" (main loop):
    read employee-master record
    at end: set end-of-file-switch "on"
    if end-of-file-switch is "off"
        if first-time-switch is "on"
            perform first-record-only-processing
            perform first-record-only-output
            set first-time-switch "off"
        else
            perform process-employee-record
            perform output-employee-results
        endif
    endif
endperform
close employee-master, report-file
stop
```

٨ - ٢٨ أسلوب القراءة الأولية مفيد بصفة خاصة فى مشاكل التحكم المتقطع control break problems ، حيث تخزن السجلات المنطقية للملف فى حقل تحكم control field واحد أو أكثر من حقل تحكم واحد . عندما تختلف القيمة الموجودة فى حقل التحكم للسجل المنطقى الحالى عن القيمة الموجودة فى حقل التحكم المتقطع للسجل المنطقى السابق ، يقال إنه حدث تحكم متقطع control break . عند هذه النقطة .. ويمكن استدعاء تشغيل خاص . مثال ذلك ، افترض أن ملف المسألة ٢٥ من هذا الفصل يشمل رقم القسم لكل عامل ، وأن الملف مرتب ترتيبًا تصاعديًا طبقًا لرقم القسم . اطبع تقريرًا للملف لايبين الرواتب الفردية فقط ، بل يبين كذلك إجمالى الراتب لكل قسم ، وإجمالى نهائى للأقسام كلها .

```
set eof "off" (end of file switch)
open files
zero: department-total, grand-total
read employee-master record
    at end: set eof "on"
if eof is "off":
    move current-department-# to previous-department-#
endif

perform until eof is "on" (main loop):
    if current-department-# not equal previous-department-#
        (this is a control break)
        perform print-department-total
        add department-total to grand-total
        move zero to department-total
```

```

        move current-department-# to previous-department-#
    endif
    perform print-current-employee-salary
    add current-employee-salary to department-total
    read employee-master record
    at end: set eof "on"
endperform

```

(end of file should be treated as a control break to
print last department total)

```

perform print-department-total
add department-total to grand-total
perform print-grand-total
close files
stop

```

فيما يلي بعض تعليقات على مسألة التحكم المتقطع فردية المستوى Single - level (حقل تحكم واحد) :

(١) لقد اختبرنا نهاية الملف في أول سجل (القراءة الأولية) . هذا غير مرغوب في حدوثه ، إلا أنه ممكن (فمثلاً يمكن حدوث أخطاء في لغة تحكم العمل ، أو خطأ من مشغل الكمبيوتر عندما ينفذ البرنامج ... إلخ) . إن وضع رقم القسم السابق في رقم القسم لأول سجل يقرأ يؤكد أن التحكم المتقطع ان ينفذ بطريقة خطأ على أول سجل عند الدخول في الدورة الرئيسية .

(٢) يجب إعادة إعداد إجمالي القسم بآئه صفر في كل مرة يتغير رقم القسم (وإلا فإنه يحدث تراكم في إجمالي القسم لإجمالي جار لكل الرواتب في كل الأقسام) .

(٣) بمجرد تمييز تغيير في حقل التحكم .. يحدث التغيير التالي في القيمة بقيمة مختلفة عن القيمة التي تسببت في التغيير الحالي . وعلى هذا .. يجب أن يتغير رقم القسم السابق إلى رقم القسم الحالي عند كل تحكم متقطع .

(٤) يمكن إيجاد الإجمالي الشامل بكفاءة أعلى عن طريق جمع إجماليات الأقسام الفردية . يجب أن يحدث هذا عند كل تحكم متقطع .

(٥) عندما تحدث نهاية الملف .. يجب أن تعامل مثل التحكم المتقطع ؛ لأنه يمكن ألا يكون هناك تغيير في حقل التحكم لآخر رقم قسم في الملف . وهذا يعني أن إجمالي آخر قسم يجب أن يطبع في نهاية الملف ، كما يجب كذلك جمع آخر إجمالي قسم على الإجمالي الشامل إذا أردنا للإجمالي الشامل أن يكون صحيحاً .

٨ - ٢٩ افترض في المسألة السابقة أن كل سجل رئيسي لعامل يحتوى على رقم وحدة division number ، وأن الملف مرتب

طبقاً لرقم الوحدة أولاً (حقل تحكم رئيسي) ، ثم طبقاً لرقم القسم داخل الوحدة (حقل تحكم صغير) . أعد تصميم

منطق المسألة رقم ٢٨ لطباعة إجمالي القسم عندما تتغير أرقام الأقسام (تحكم متقطع صغير) ، وطباعة إجمالي الوحدة

عندما يتغير رقم الوحدة (تحكم متغير رئيسي) . يجب أن يأخذ التقرير في الشكل الموجود في شكل ٨ - ٨ .

```

employee salary (division 1000, department 1)
employee salary (division 1000, department 1)
department 1 total
employee salary (division 1000, department 2)
employee salary (division 1000, department 2)
department 2 total
division 1000 total
employee salary (division 2000, department 1)
department 1 total
employee salary (division 2000, department 2)
employee salary (division 2000, department 2)
department 2 total
division 2000 total
grand total

```

شكل (٨ - ٨)

```

set eof "off"
open files
zero: department-total, division-total, grand-total
read employee-master record
    at end: set eof "on"
if eof is "off":
    move current-department-# to previous-department-#
    move current-division-# to previous-division-#
endif

perform until eof is "on" (main loop):
    if current-division-# not equal previous-division-#
        (this is a major control break)

        perform handle-department-break
        perform print-division-total
        add division-total to grand-total
        move zero to division-total
        move current-division-# to previous-division-#
    else if current-department-# not equal previous-department-#
        (this is a minor control break)
        perform handle-department-break
    endif
    perform print-current-employee-salary
    add current-employee-salary to department-total
    read employee-master record
    at end: set eof "on"
endperform

(end of file should be treated as a major control break)

perform handle-department-break
perform print-division-total
add division-total to grand-total
perform print-grand-total
close files
stop

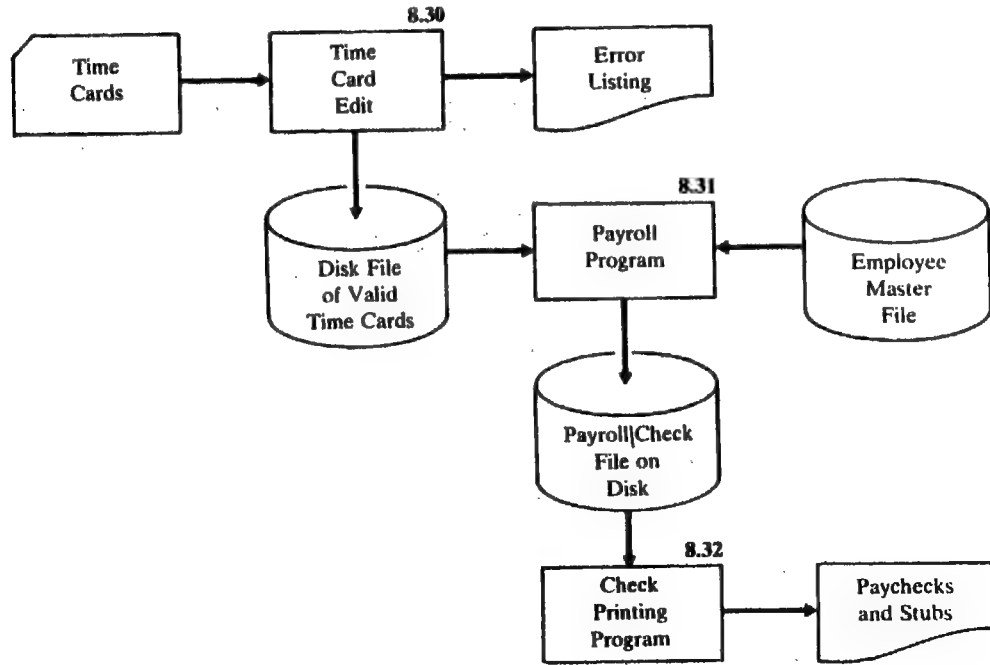
```

هنا يعرف break - department - handle كما يلي:

```
handle-department-break:
perform print-department-total
add department-total to division-total
move zero to department-total
move current-department-# to previous-department-#
```

تعليقات على خوارزمي تحكم متقطع مزيج المستوى :

- ١ - يجب وضع قيم ابتدائية لحقل التحكم الصغير ، وحقل التحكم الرئيسي مساوية لقيم أول سجل منطقي . يعد تحقيق هذا بسيطاً اذا استخدمت قراءة أولية .
- ٢ - يبدأ منطق دورة البرنامج الرئيسية باختيار التحكم المتقطع الرئيسي . لاحظ أنه بسبب وضع قيمة ابتدائية لرقم الوحدة السابقة بأنها أول رقم وحدة حالية ، فلا يتحدد تحكم متقطع لأول سجل في الملف .
- ٣ - يبدأ تشغيل التحكم المتقطع الرئيسي بعمل تحكم متقطع صغير . ومن الأخطاء الشائعة الحدوث للمبتدئين اهمالهم حقيقة أنه عند تغيير حقل تحكم رئيسي (رقم الوحدة) يجب أن تطبع إجماليات حقل التحكم الصغير (رقم القسم) الأخير ، ويعاد إعداد إجمالي حقل التحكم الصغير . افحص شكل ٨ - ٨ للتأكد من فهمك سبب ذلك .
- ٤ - بعد معالجة تشغيل التحكم المتقطع الرئيسي للتحكم المتقطع الصغير ، فيؤدي جزء التقطع الرئيسي كل ما هو مناسب للتغيير في حقل التحكم الرئيسي . لاحظ بصفة خاصة أن رقم الوحدة السابق أعيد إعداده بوضع رقم الوحدة الحالي فيه أثناء تشغيل التحكم المتقطع الرئيسي ، وأعيدت إجماليات حقل التحكم الرئيسي لتصبح صفراً .
- ٥ - بعد أن تختبر الدورة الرئيسية للتحكم المتقطع الرئيسي .. فيجب أن تختبر للتحكم المتقطع الصغير . لاحظ أن ذلك يمكن أن يحدث بهيكل IF خطي . ينفذ جزء تقطع صغير مثل مقطع PERFORM ، حيث إنه يجب استدعاؤه من جزء التقطع الرئيسي ، وجزء التقطع الصغير .
- ٦ - منطق كل من جزئي التحكم المتقطع هو كما يلي : (١) استدع جزء التحكم المتقطع المنخفض المستوى التالي (اذا كان هناك مثل هذا التحكم المتقطع) ، (٢) اطبع إجماليات تحكم لهذا التقطع ، (٣) ضف إجمالي التحكم هذا إلى إجمالي المستوى الأعلى التالي (٤) ضع صفراً في هذا الإجمالي (٥) ضع حقل التحكم الحالي في حقل التحكم السابق .
- ٧ - مرة أخرى .. يجب أن تعامل نهاية الملف كتحكم متقطع رئيسي (أولاً فإن إجمالي آخر قسم وإجمالي آخر وحدة لن يطبعاً ولن يضافاً إلى الإجمالي الشامل) .
- ٨ - شكل ٢٠ - ٨ هو خريطة مسار النظام system flowchart - والتي ترسم العلاقات بين البرامج والملفات - لنظام رواتب (مبسطة جداً) . (رموز ملف البطاقات وملف القرص واضحة ذاتياً ، وملف الطباعة يبدو كورقة مقطوعة) . صمم واكتب شفرة برنامج تنقيح واختبره لنظام الرواتب هذا .



شكل (٨ - ٩)

كما نعرف (مثال ٦ - ٢٥) الغرض من برنامج التنقيح هو التأكيد من صحة المعلومات التي يقوم بإدخالها الأدميون .

يجب أن يعطى برنامج التنقيح بصفة خاصة أجوبة على الأسئلة التالية :

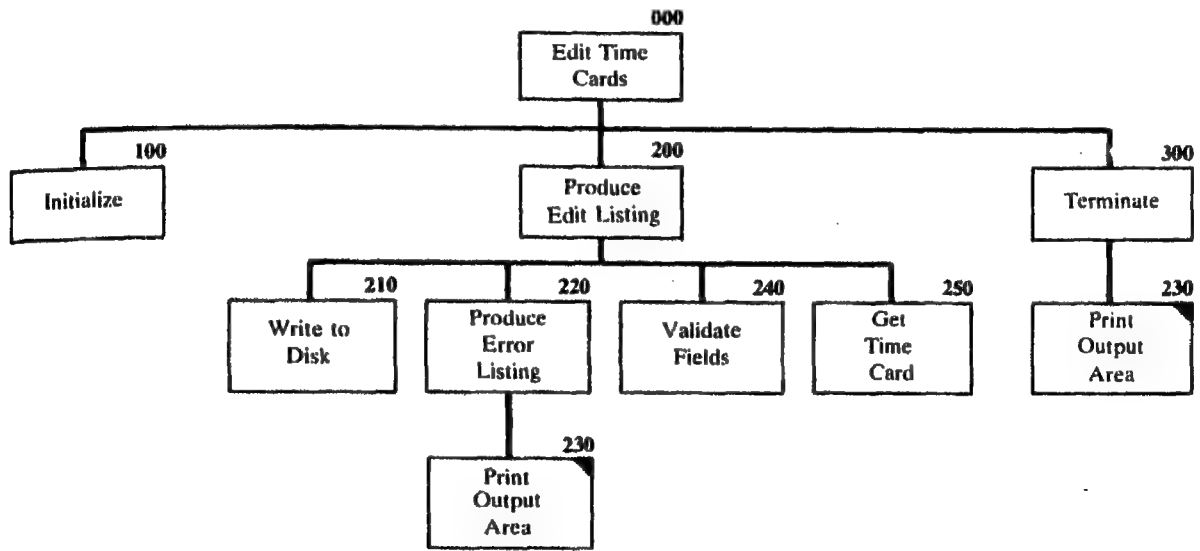
- ١ - هل الحقول العددية عديدة فعلا ؟ وهل الحقول الحرفية حرفية .. فعلا ؟ ... إلخ .
- ٢ - هل تقع القيم داخل مدى مناسب (مثل اليوم من الشهر يجب أن يقع بين 1 و 31) ؟
- ٣ - هل القيم الموجودة في السجل متوافقة مع بعضه البعض (مثل أن تاريخ الاستلام لا يمكن أن يسبق تاريخ الشحن) ؟
- ٤ - هل تحتوي الحقول المستخدم فيها رموزاً على الرموز المسموح بها فقط ؟
- ٥ - هل السجلات المنطقية التي يفترض أنها مخزنة (مرتبة على التتابع) في حقل خاص مرتبة ترتيباً صحيحاً ؟ (هذا النوع من التأكد من الصحة عادة ما يسمى بالتأكد من التتابع) .

٦ - هل إجماليات تحكم الدفعة دقيقة ؟ (حيث إن دفعات من السجلات يقوم بإدخالها مشغل الكمبيوتر ، فعادة ما تحسب إجماليات تحكم يدوي ، بجمع كل محتويات حقول محددة . عندما يقوم برنامج كمبيوتر بتشغيل الملف الناتج ، فيمكن أن يجمع إجمالياً لنفس الحقل (الحقول) كذلك . إذا اتفق إجمالي الكمبيوتر مع الإجمالي اليدوي ، فيفترض أن كل البيانات أجرى عليها تشغيل) .

نرسم الآن خريطة هيكل (شكل ٨ - ١٠) ، وخريطة هيبو (شكل ٨ - ١١) ، وتنفيذ أول وثاني اختبار من القمة إلى القاعدة (شكل ٨ - ١٢ وشكل ٨ - ١٣) ، وآخر تنفيذ (شكل ٨ - ١٤) ، لبرنامج تنقيح الرواتب .

لاحظ بصفة خاصة معاملة عمل الصفحات بدون جزء LINAGE (انظر الأجزاء ، AREA - OUTPUT - PRINT - 230)

(100 - INITIALIZES)



شكل (٨ - ١)

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>(XX)</u>	Module Name <u>EDIT-TIME-CARDS</u>	

Input	Process	Output
ws-end-time-file-sw	<ol style="list-style-type: none"> 1. perform initialize 2. perform produce-edit-listing until end of time card file 3. perform terminate 4. stop 	

شكل (٨ - ١) صفحة ١

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>100</u>	Module Name <u>INITIALIZE</u>	

Input	Process	Output
ws-page-size, system date	<ol style="list-style-type: none"> 1. open files 2. move "no" to ws-end-time-file-sw 3. zero: ws-page-number, ws-number-employees, ws-total-hours 4. move ws-page-size to ws-lines-on-page 5. get system date and move to heading 	ws-end-time-file-sw, ws-page-number, ws-number-employees, ws-total-hours, ws-lines-on-page, heading

شكل (٨ - ١) صفحة ٢

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>200</u>	Module Name <u>PRODUCE-EDIT-LISTING</u>	

Input	Process	Output
ws-end-time- file-sw, ws- error- detected-sw, time-card- hours	<ol style="list-style-type: none"> perform get-time-card if a card was read <ul style="list-style-type: none"> perform validate-fields if record was valid <ul style="list-style-type: none"> add 1 to ws-number-employees add time-card-hours to ws-total-hours perform write-to-disk else <ul style="list-style-type: none"> perform produce-error-listing 	ws-number- employees, ws-total-hours

شكل (٨ - ١١) صفحة ٣

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>300</u>	Module Name <u>TERMINATE</u>	

Input	Process	Output
ws-number- employees, ws-total hours	<ol style="list-style-type: none"> move ws-number-employees and ws-total-hours to footing-line perform print-output-area close files 	footing-line

شكل (٨ - ١١) صفحة ٤

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>220</u>	Module Name <u>PRODUCE ERROR LISTING</u>	

Input	Process	Output
time-card, underline	<ol style="list-style-type: none"> 1. move all time card fields to detail-line 2. move detail-line to output-area 3. perform print-output-area 4. move underline to output-area 5. perform print-output-area 	detail-line output-area

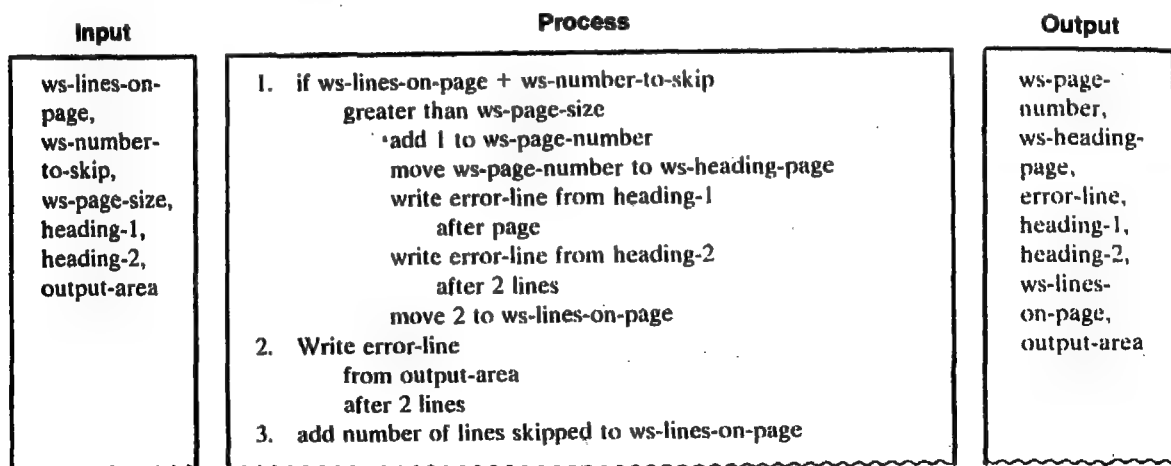
شكل (٨ - ١١) صفحة ٥

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>240</u>	Module Name <u>VALIDATE-FIELDS</u>	

Input	Process	Output
time-card-id, time-card- hours, time-card- date, time-card- department	<ol style="list-style-type: none"> 1. move "no" to error-detected-sw 2. move spaces to under-line 3. if time-card-id not numeric set up underline for id set error-detected-sw to "yes" 4. if time-card-hours not numeric set up underline for hours set error-detected-sw to "yes" 5. if time-card-date not numeric set up underline for date set error-detected-sw to "yes" 6. if time-card-department not numeric set up underline for department set error-detected-sw to "yes" 	error- detected-sw, underline

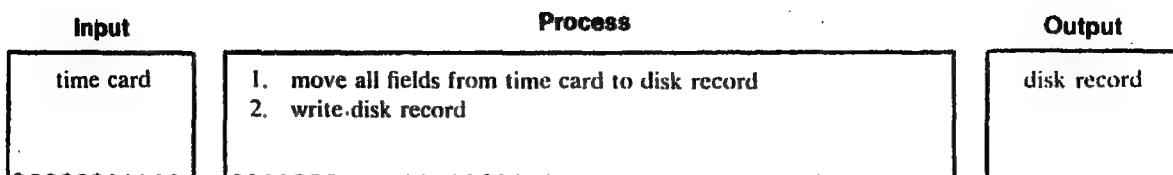
شكل (٨ - ١١) صفحة ٦

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>230</u>	Module Name <u>PRINT-OUTPUT-AREA</u>	



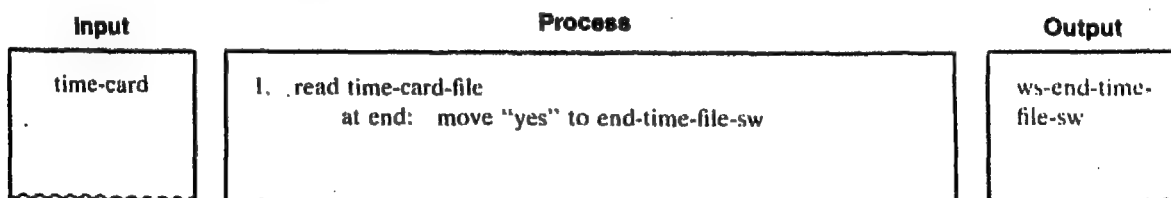
شكل (٨ - ١١) صفحة ٧

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>210</u>	Module Name <u>WRITE-TO-DISK</u>	



شكل (٨ - ١١) صفحة ٨

Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>250</u>	Module Name <u>GET-TIME-CARD</u>	



شكل (٨ - ١١) صفحة ٩

```

00001      IDENTIFICATION DIVISION.
00002
00003      PROGRAM-ID.    TIMEEDIT.
00004
00005      AUTHOR.    LARRY NEWCOMER.
00006      INSTALLATION.  PENN STATE UNIVERSITY -- YORK CAMPUS.
00007
00008      ENVIRONMENT DIVISION.
00009
00010      CONFIGURATION SECTION.
00011      SOURCE-COMPUTER.  IBM-3081 WITH DEBUGGING MODE.
00012      OBJECT-COMPUTER.  IBM-3081.
00013
00014      INPUT-OUTPUT SECTION.
00015      FILE-CONTROL.
00016
00017          SELECT TIME-CARD-FILE
00018              ASSIGN TO TIMECARD
00019              ORGANIZATION IS SEQUENTIAL
00020              ACCESS IS SEQUENTIAL
00021
00022          SELECT VALID-TIME-FILE
00023              ASSIGN TO DISKTIME
00024              ORGANIZATION IS SEQUENTIAL
00025              ACCESS IS SEQUENTIAL
00026
00027          SELECT ERROR-LISTING-FILE
00028              ASSIGN TO ERRORLOG
00029              ORGANIZATION IS SEQUENTIAL
00030              ACCESS IS SEQUENTIAL.
00031
00032
00033      DATA DIVISION.
00034
00035      FILE SECTION.
00036
00037      FD  TIME-CARD-FILE
00038          RECORD CONTAINS 80 CHARACTERS
00039          LABEL RECORDS ARE OMITTED
00040
00041
00042      01  TIME-CARD-RECORD.
00043          05  TIME-CARD-ID                PIC X(5).
00044          05  TIME-CARD-HOURS             PIC 9(2)V9.
00045          05  TIME-CARD-X-HOURS           REDEFINES TIME-CARD-HOURS
00046                                          PIC X(3).
00047          05  TIME-CARD-CLOSING-DATE      PIC X(6).
00048          05  TIME-CARD-DEPARTMENT        PIC X(4).
00049          05  FILLER                     PIC X(62).
00050
00051      FD  VALID-TIME-FILE
00052          BLOCK CONTAINS 0 RECORDS
00053          RECORD CONTAINS 23 CHARACTERS
00054          LABEL RECORDS ARE STANDARD
00055
00056
00057      01  VALID-TIME-RECORD.
00058          05  VALID-TIME-ID                PIC X(5).
00059          05  VALID-TIME-HOURS             PIC S9(2)V9  COMP-3.
00060          05  VALID-TIME-CLOSING-DATE      PIC X(6).

```

```

00061      05 VALID-TIME-DEPARTMENT    PIC X(4).
00062      05 VALID-TIME-EDIT-DATE     PIC X(6).
00063
00064      FD ERROR-LISTING-FILE
00065      RECORD CONTAINS 132 CHARACTERS
00066      LABEL RECORDS ARE OMITTED
00067
00068
00069      01 ERROR-REPORT-LINE            PIC X(132).
00070
00071      WORKING-STORAGE SECTION.
00072
00073      01 PROGRAM-SWITCHES.
00074          05 WS-END-TIME-FILE-SW      PIC X(3).
00075              88 NO-MORE-RECORDS      VALUE "YES".
00076              88 RECORD-AVAILABLE     VALUE "NO".
00077          05 WS-ERROR-DETECTED-SW     PIC X(3).
00078              88 VALID-RECORD         VALUE "NO ".
00079              88 INVALID-RECORD       VALUE "YES".
00080
00081      01 PROGRAM-COUNTERS.
00082          05 WS-PAGE-NUMBER            PIC S9(3)      COMP-3.
00083          05 WS-NUMBER-EMPLOYEES       PIC S9(5)      COMP-3.
00084          05 WS-NUMBER-TO-SKIP         PIC S9          COMP SYNC.
00085          05 WS-LINES-ON-PAGE          PIC S9(2)      COMP SYNC.
00086
00087      01 PROGRAM-TOTAL-AREAS.
00088          05 WS-TOTAL-HOURS            PIC S9(7)V9     COMP-3.
00089
00090      01 PROGRAM-CONSTANTS.
00091          05 WS-PAGE-SIZE               PIC S9(2)      COMP SYNC
00092          VALUE +50.
00093
00094      01 WS-HEADING-LINE-1.
00095          05 FILLER                    PIC X(5)        VALUE SPACES.
00096          05 FILLER                    PIC X(15)       VALUE "TIME CARD EDIT".
00097
00098          05 WS-HEADING-MM             PIC Z9.
00099          05 FILLER                    PIC X           VALUE "/".
00100          05 WS-HEADING-DD             PIC 99.
00101          05 FILLER                    PIC X           VALUE "/".
00102          05 WS-HEADING-YY             PIC 99.
00103          05 FILLER                    PIC X(3)        VALUE SPACES.
00104          05 FILLER                    PIC X(5)        VALUE "PAGE ".
00105          05 WS-HEADING-PAGE           PIC ZZ9.
00106          05 FILLER                    PIC X(93)       VALUE SPACES.
00107
00108      01 WS-HEADING-LINE-2.
00109          05 FILLER                    PIC X(7)        VALUE " ID".
00110          05 FILLER                    PIC X(5)        VALUE "HRS".
00111          05 FILLER                    PIC X(8)        VALUE " DATE".
00112          05 FILLER                    PIC X(112)       VALUE "DEPT".
00113
00114      01 WS-DETAIL-LINE.
00115          05 WS-DETAIL-ID              PIC X(5).
00116          05 FILLER                    PIC X(2)        VALUE SPACES.

```

```

00117      05 WS-DETAIL-HOURS      PIC X(3).
00118      05 FILLER                PIC X(2)      VALUE SPACES.
00119      05 WS-DETAIL-DATE        PIC X(6).
00120      05 FILLER                PIC X(2)      VALUE SPACES.
00121      05 WS-DETAIL-DEPARTMENT  PIC X(4).
00122      05 FILLER                PIC X(108)     VALUE SPACES.
00123
00124      01 WS-UNDER-LINE.
00125      05 WS-UNDER-ID          PIC X(5).
00126      05 FILLER                PIC X(2)      VALUE SPACES.
00127      05 WS-UNDER-HOURS       PIC X(3).
00128      05 FILLER                PIC X(2)      VALUE SPACES.
00129      05 WS-UNDER-DATE        PIC X(6).
00130      05 FILLER                PIC X(2)      VALUE SPACES.
00131      05 WS-UNDER-DEPARTMENT  PIC X(4).
00132      05 FILLER                PIC X(108)     VALUE SPACES.
00133
00134      PROCEDURE DIVISION.
00135
00136      000-EDIT-TIME-CARDS.
00137
00138          PERFORM 100-INITIALIZE
00139          PERFORM 200-PRODUCE-EDIT-LISTING
00140              UNTIL NO-MORE-RECORDS
00141          PERFORM 300-TERMINATE
00142          STOP RUN
00143
00144
00145      100-INITIALIZE.
00146
00147      D      DISPLAY "100 ENTERED"
00148      .
00149      200-PRODUCE-EDIT-LISTING.
00150
00151      D      DISPLAY "200 ENTERED"
00152      D      MOVE "YES" TO WS-END-TIME-FILE-SW
00153      .
00154
00155      300-TERMINATE.
00156
00157      D      DISPLAY "300 ENTERED"
00158      .

```

شكل (٨ - ١٢) تكملة


```

00153      PROCEDURE DIVISION.
00154
00155      000-EDIT-TIME-CARDS.
00156
00157          PERFORM 100-INITIALIZE
00158          PERFORM 200-PRODUCE-EDIT-LISTING
00159              UNTIL NO-MORE-RECORDS
00160          PERFORM 300-TERMINATE
00161          STOP RUN
00162
00163
00164      100-INITIALIZE.
00165
00166      D      DISPLAY "100 ENTERED"
00167          OPEN      INPUT      TIME-CARD-FILE
00168                  OUTPUT      VALID-TIME-FILE
00169                      ERROR-LISTING-FILE
00170          MOVE "NO" TO      WS-END-TIME-FILE-SW
00171          MOVE ZERO TO      WS-PAGE-NUMBER
00172                      WS-NUMBER-EMPLOYEES
00173                      WS-TOTAL-HOURS
00174          MOVE WS-PAGE-SIZE TO      WS-LINES-ON-PAGE
00175          ACCEPT WS-SYSTEM-DATE FROM DATE
00176          MOVE SYSTEM-YY TO      WS-HEADING-YY
00177          MOVE SYSTEM-MM TO      WS-HEADING-MM
00178          MOVE SYSTEM-DD TO      WS-HEADING-DD
00179
00180
00181      200-PRODUCE-EDIT-LISTING.
00182
00183      D      DISPLAY "200 ENTERED"
00184          PERFORM 250-GET-TIME-CARD
00185          IF RECORD-AVAILABLE
00186              PERFORM 240-VALIDATE-FIELDS
00187              IF VALID-RECORD .
00188                  PERFORM 210-WRITE-TO-DISK
00189              ELSE
00190                  PERFORM 220-PRODUCE-ERROR-LISTING
00191
00192
00193      210-WRITE-TO-DISK.
00194
00195      D      DISPLAY "210 ENTERED"
00196
00197
00198      220-PRODUCE-ERROR-LISTING.
00199
00200      D      DISPLAY "220 ENTERED"
00201
00202
00203      230-PRINT-OUTPUT-AREA.
00204
00205      D      DISPLAY "230 ENTERED"
00206
00207
00208      240-VALIDATE-FIELDS.
00209
00210      D      DISPLAY "240 ENTERED"
00211      D      ON 1 AND EVERY 2

```

```

00212      D      MOVE "YES" TO WS-ERROR-DETECTED-SW
00213      D      ELSE
00214      D      MOVE "NO" TO WS-ERROR-DETECTED-SW
00215
00216
00217      250-GET-TIME-CARD.
00218
00219      READ TIME-CARD-FILE
00220      AT END
00221      MOVE "YES" TO WS-END-TIME-FILE-SW
00222
00223
00224      300-TERMINATE.
00225
00226      D      DISPLAY "300 ENTERED"
00227      MOVE WS-NUMBER-EMPLOYEES      TO WS-FOOTING-COUNT
00228      MOVE WS-TOTAL-HOURS          TO WS-FOOTING-TOTAL
00229      MOVE WS-FOOTING-LINE          TO WS-OUTPUT-AREA
00230      MOVE WS-SKIP-BEFORE-FOOTING TO WS-NUMBER-TO-SKIP
00231      MOVE WS-FOOTING-LINE          TO WS-OUTPUT-AREA
00232      PERFORM 230-PRINT-OUTPUT-AREA
00233
00234      CLOSE    TIME-CARD-FILE
00235              VALID-TIME-FILE
00236              ERROR-LISTING-FILE
00237

```

شكل (٨ - ١٣)

```

00001      IDENTIFICATION DIVISION.
00002
00003      PROGRAM-ID.    TIMEEDIT.
00004
00005      AUTHOR.    LARRY NEWCOMER.
00006      INSTALLATION.    PENN STATE UNIVERSITY -- YORK CAMPUS.
00007
00008      ENVIRONMENT DIVISION.
00009
00010      CONFIGURATION SECTION.
00011      SOURCE-COMPUTER.    IBM-3081 WITH DEBUGGING MODE.
00012      OBJECT-COMPUTER.    IBM-3081.
00013
00014      INPUT-OUTPUT SECTION.
00015      FILE-CONTROL.
00016
00017      SELECT TIME-CARD-FILE
00018      ASSIGN TO TIMECARD
00019      ORGANIZATION IS SEQUENTIAL
00020      ACCESS IS SEQUENTIAL
00021
00022      SELECT VALID-TIME-FILE
00023      ASSIGN TO DISKTIME
00024      ORGANIZATION IS SEQUENTIAL

```

شكل (٨ - ١٤)

```

00025             ACCESS IS SEQUENTIAL
00026
00027             SELECT ERROR-LISTING-FILE
00028             ASSIGN TO ERRORLOG
00029             ORGANIZATION IS SEQUENTIAL
00030             ACCESS IS SEQUENTIAL
00031
00032
00033             DATA DIVISION.
00034
00035             FILE SECTION.
00036
00037             FD  TIME-CARD-FILE
00038             RECORD CONTAINS 80 CHARACTERS
00039             LABEL RECORDS ARE OMITTED
00040
00041
00042             01  TIME-CARD-RECORD.
00043                 05  TIME-CARD-ID             PIC X(5).
00044                 05  TIME-CARD-HOURS          PIC 9(2)V9.
00045                 05  TIME-CARD-X-HOURS        REDEFINES TIME-CARD-HOURS
00046                                     PIC X(3).
00047                 05  TIME-CARD-CLOSING-DATE    PIC X(6).
00048                 05  TIME-CARD-DEPARTMENT      PIC X(4).
00049                 05  FILLER                   PIC X(62).
00050
00051             FD  VALID-TIME-FILE
00052             BLOCK CONTAINS 0 RECORDS
00053             RECORD CONTAINS 23 CHARACTERS
00054             LABEL RECORDS ARE STANDARD
00055
00056
00057             01  VALID-TIME-RECORD.
00058                 05  VALID-TIME-ID             PIC X(5).
00059                 05  VALID-TIME-HOURS          PIC S9(2)V9 COMP-3.
00060                 05  VALID-TIME-CLOSING-DATE    PIC X(6).
00061                 05  VALID-TIME-DEPARTMENT      PIC X(4).
00062                 05  VALID-TIME-EDIT-DATE      PIC X(6).
00063
00064             FD  ERROR-LISTING-FILE
00065             RECORD CONTAINS 132 CHARACTERS
00066             LABEL RECORDS ARE OMITTED
00067
00068
00069             01  ERROR-REPORT-LINE             PIC X(132).
00070
00071             WORKING-STORAGE SECTION.
00072
00073             01  PROGRAM-SWITCHES.
00074                 05  WS-END-TIME-FILE-SW        PIC X(3).
00075                 88  NO-MORE-RECORDS            VALUE "YES".
00076                 88  RECORD-AVAILABLE          VALUE "NO".
00077                 05  WS-ERROR-DETECTED-SW       PIC X(3).
00078                 88  VALID-RECORD              VALUE "NO ".
00079                 88  INVALID-RECORD            VALUE "YES".

```

شكل (٨ - ١٤) تكملة

```

00080
00081      01 PROGRAM-COUNTERS.
00082      05 WS-PAGE-NUMBER          PIC S9(3)      COMP-3.
00083      05 WS-NUMBER-EMPLOYEES     PIC S9(5)      COMP-3.
00084      05 WS-NUMBER-TO-SKIP       PIC S9         COMP SYNC.
00085      05 WS-LINES-ON-PAGE        PIC S9(2)      COMP SYNC.
00086
00087      01 PROGRAM-TOTAL-AREAS.
00088      05 WS-TOTAL-HOURS          PIC S9(7)V9     COMP-3.
00089
00090      01 PROGRAM-CONSTANTS.
00091      05 WS-PAGE-SIZE             PIC S9(2)      COMP SYNC
00092      05 WS-SKIP-BEFORE-HEADING  PIC S9(2)      COMP SYNC
00093      05 WS-SKIP-BEFORE-FOOTING  PIC S9(2)      COMP SYNC
00094      05 WS-SKIP-BEFORE-DETAIL   PIC S9(2)      COMP SYNC
00095      05 WS-SKIP-BEFORE-UNDER    PIC S9(2)      COMP SYNC
00096      05 WS-SKIP-BEFORE-UNDER    PIC S9(2)      COMP SYNC
00097      05 WS-SKIP-BEFORE-UNDER    PIC S9(2)      COMP SYNC
00098      05 WS-SKIP-BEFORE-UNDER    PIC S9(2)      COMP SYNC
00099      05 WS-SKIP-BEFORE-UNDER    PIC S9(2)      COMP SYNC
00100      05 WS-SKIP-BEFORE-UNDER    PIC S9(2)      COMP SYNC
00101
00159      PROCEDURE DIVISION.
00160
00161      000-EDIT-TIME-CARDS.
00162
00163          PERFORM 100-INITIALIZE
00164          PERFORM 200-PRODUCE-EDIT-LISTING
00165          UNTIL NO-MORE-RECORDS
00166          PERFORM 300-TERMINATE
00167          STOP RUN
00168
00169
00170      100-INITIALIZE.
00171
00172      D    DISPLAY "100 ENTERED"
00173      OPEN  INPUT  TIME-CARD-FILE
00174           OUTPUT VALID-TIME-FILE
00175           ERROR-LISTING-FILE
00176      MOVE "NO" TO WS-END-TIME-FILE-SW
00177      MOVE ZERO TO WS-PAGE-NUMBER
00178           WS-NUMBER-EMPLOYEES
00179           WS-TOTAL-HOURS
00180      MOVE WS-PAGE-SIZE TO WS-LINES-ON-PAGE
00181      ACCEPT WS-SYSTEM-DATE FROM DATE
00182      MOVE SYSTEM-YY TO WS-HEADING-YY
00183      MOVE SYSTEM-MM TO WS-HEADING-MM
00184      MOVE SYSTEM-DD TO WS-HEADING-DD
00185
00186
00187      200-PRODUCE-EDIT-LISTING.
00188
00189      D    DISPLAY "200 ENTERED"
00190      PERFORM 250-GET-TIME-CARD
00191      IF RECORD-AVAILABLE
00192          PERFORM 240-VALIDATE-FIELDS
00193      IF VALID-RECORD
00194          ADD 1 TO WS-NUMBER-EMPLOYEES
00195          ADD TIME-CARD-HOURS TO WS-TOTAL-HOURS

```

شكل (٨ - ١٤) تكملة

```

00196             PERFORM 210-WRITE-TO-DISK
00197             ELSE
00198             PERFORM 220-PRODUCE-ERROR-LISTING
00199
00200
00201         210-WRITE-TO-DISK.
00202
00203     D   DISPLAY "210 ENTERED"
00204         MOVE TIME-CARD-ID             TO VALID-TIME-ID
00205         MOVE TIME-CARD-HOURS          TO VALID-TIME-HOURS
00206         MOVE TIME-CARD-CLOSING-DATE   TO VALID-TIME-CLOSING-DATE
00207         MOVE TIME-CARD-DEPARTMENT     TO VALID-TIME-DEPARTMENT
00208         MOVE WS-SYSTEM-DATE           TO VALID-TIME-EDIT-DATE
00209         WRITE VALID-TIME-RECORD
00210
00211
00212         220-PRODUCE-ERROR-LISTING.
00213
00214     D   DISPLAY "220 ENTERED"
00215         MOVE TIME-CARD-ID             TO WS-DETAIL-ID
00216         MOVE TIME-CARD-X-HOURS        TO WS-DETAIL-HOURS
00217         MOVE TIME-CARD-CLOSING-DATE   TO WS-DETAIL-DATE
00218         MOVE TIME-CARD-DEPARTMENT     TO WS-DETAIL-DEPARTMENT
00219
00220         MOVE WS-DETAIL-LINE           TO WS-OUTPUT-AREA
00221         MOVE WS-SKIP-BEFORE-DETAIL    TO WS-NUMBER-TO-SKIP
00222         PERFORM 230-PRINT-OUTPUT-AREA
00223
00224         MOVE WS-UNDER-LINE           TO WS-OUTPUT-AREA
00225         MOVE WS-SKIP-BEFORE-UNDER     TO WS-NUMBER-TO-SKIP
00226         PERFORM 230-PRINT-OUTPUT-AREA
00227
00228
00229         230-PRINT-OUTPUT-AREA.
00230
00231     D   DISPLAY "230 ENTERED"
00232         IF WS-LINES-ON-PAGE + WS-NUMBER-TO-SKIP
00233             GREATER THAN WS-PAGE-SIZE
00234             ADD 1 TO WS-PAGE-NUMBER
00235             MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE
00236             WRITE ERROR-REPORT-LINE
00237                 FROM WS-HEADING-LINE-1
00238                 AFTER ADVANCING PAGE
00239             WRITE ERROR-REPORT-LINE
00240                 FROM WS-HEADING-LINE-2
00241             AFTER ADVANCING WS-SKIP-BEFORE-HEADING LINES
00242             MOVE WS-SKIP-BEFORE-HEADING TO WS-LINES-ON-PAGE
00243
00244             WRITE ERROR-REPORT-LINE
00245                 FROM WS-OUTPUT-AREA
00246             AFTER ADVANCING WS-NUMBER-TO-SKIP LINES
00247             ADD WS-NUMBER-TO-SKIP TO WS-LINES-ON-PAGE
00248
00249
00250         240-VALIDATE-FIELDS.
00251
00252     D   DISPLAY "240 ENTERED"
00253         MOVE "NO" TO WS-ERROR-DETECTED-SW
00254         MOVE SPACES TO WS-UNDER-LINE
00255         IF TIME-CARD-ID NOT NUMERIC
00256             MOVE ALL "-" TO WS-UNDER-ID
00257             MOVE "YES" TO WS-ERROR-DETECTED-SW

```

```

00258
00259      IF TIME-CARD-HOURS NOT NUMERIC
00260          MOVE ALL "-" TO WS-UNDER-HOURS
00261          MOVE "YES" TO WS-ERROR-DETECTED-SW
00262
00263      IF TIME-CARD-CLOSING-DATE NOT NUMERIC
00264          MOVE ALL "-" TO WS-UNDER-DATE
00265          MOVE "YES" TO WS-ERROR-DETECTED-SW
00266
00267      IF TIME-CARD-DEPARTMENT NOT NUMERIC
00268          MOVE ALL "-" TO WS-UNDER-DEPARTMENT
00269          MOVE "YES" TO WS-ERROR-DETECTED-SW
00270
00271
00272      250-GET-TIME-CARD.
00273
00274          READ TIME-CARD-FILE
00275          AT END
00276              MOVE "YES" TO WS-END-TIME-FILE-SW
00277
00278
00279      300-TERMINATE.
00280
00281      D      DISPLAY "300 ENTERED"
00282          MOVE WS-NUMBER-EMPLOYEES TO WS-FOOTING-COUNT
00283          MOVE WS-TOTAL-HOURS TO WS-FOOTING-TOTAL
00284          MOVE WS-FOOTING-LINE TO WS-OUTPUT-AREA
00285          MOVE WS-SKIP-BEFORE-FOOTING TO WS-NUMBER-TO-SKIP
00286          MOVE WS-FOOTING-LINE TO WS-OUTPUT-AREA
00287          PERFORM 230-PRINT-OUTPUT-AREA
00288
00289          CLOSE TIME-CARD-FILE
00290          VALID-TIME-FILE
00291          ERROR-LISTING-FILE
00292

```

```

TIME CARD EDIT 5/09/83 PAGE 1
ID HRS DATE DEPT
1 111 111 111111 1111
-----
11 11 111 111111 1111
-----
111 1 111 111111 1111
-----
11111 11 111111 1111
-----
11111 1 1 111111 1111
-----
.
.
.
11111 111 111111 11 1
-----

```

```

TIME CARD EDIT 5/09/83 PAGE 2
ID HRS DATE DEPT
11111 111 111111 111
-----
-----
-----
-----

```

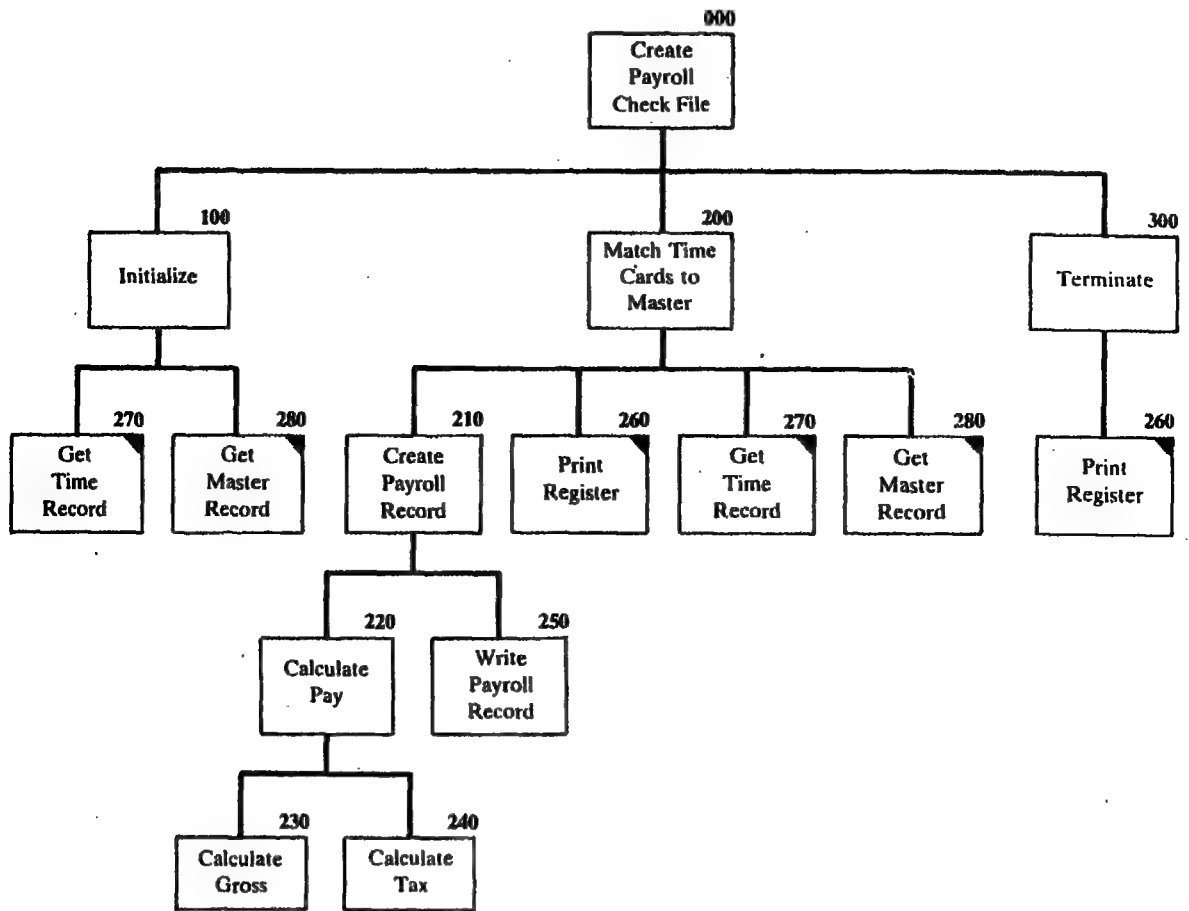
EMPLOYEES= 2 TOTAL HOURS= 33.3

شكل (٨ - ١٤) تكملة

المسألة رقم ٢٢ من الفصل الحادى عشر تعزز الحل السابق .

٨ - ٢١ ارسم خريطة هيكل لبرنامج رواتب المسألة السابقة .

انظر شكل ٨ - ١٥ . قلب الخوارزمى هو الجزء الذى يجعل السجل المنطقى لبطاقه وقت صحيحة (من ملف قرص يتم إنتاجه بواسطة برنامج المسألة السابقه) متفقا مع السجلات المنطقية المناظرة من ملف العاملين الرئيسى ، والذى يعطى معدل الأجر لكل عامل . بافتراض أن كلاً من الملفين مرتب ترتيباً تصاعدياً ، طبقاً لرقم تعريف العامل ، فيمكن عمل خريطة هيرو لهذا الجزء كما تبدو فى شكل ٨ - ١٦ (انظر المسألة رقم ٢٣ فى الفصل الحادى عشر لرؤية برنامج الكويل كاملاً .)



شكل (٨ - ١٥)

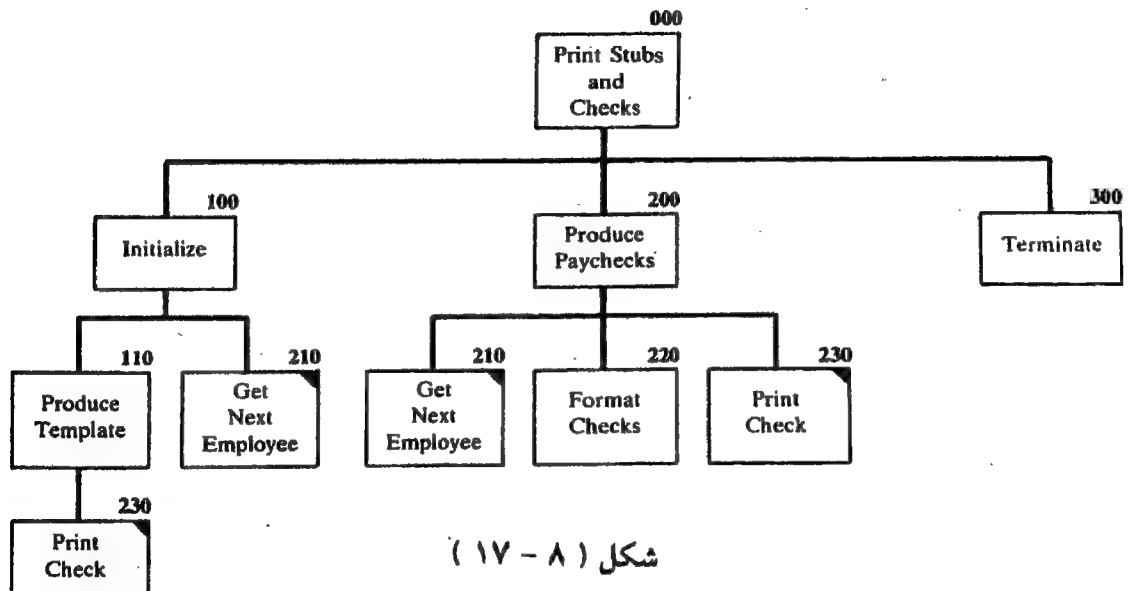
Designer <u>L. Newcomer</u>	System/Program Name <u>PAYROLL</u>	Date <u>10/83</u>
Module No. <u>200</u>	Module Name <u>MATCH-TIME-CARDS-TO-MASTER</u>	

Input	Process	Output
time-id, master-id	<ol style="list-style-type: none"> 1. set detail-line to spaces 2. if time-id = master-id <ul style="list-style-type: none"> perform 210-create-payroll-record perform 270-get-time-record perform 280-get-master-record else if time-id less than master-id <ul style="list-style-type: none"> format "unmatched time record" error perform 270-get-time-record else <ul style="list-style-type: none"> format "inactive employee" message (<i>unmatched master</i>) perform 280-get-master-record endif 3. perform 260-print-register 	detail-line, message-area

شكل (٨ - ١٦)

٨ - ٣٢ ارسم خريطة هيكل واختبر طباعة البرنامج لنظام رواتب المسالتين ٢٠ و ٣١ السابقين .

انظر شكل ٨ - ١٧ ... يمكن استخدام عبارة ACCEPT في مقطع الكويل INITIALIZE - 100 ، وذلك للحصول على تاريخ اليوم ، والتأكد من الرقم الذى سيطبع على أول شيك ، وتنتج أرقام الشيكات التالية باضافة 1 لكل شيك . (يعطى مثال ١٠ فى الفصل الثانى عشر البرنامج كاملا بلغة الكويل .)



شكل (٨ - ١٧)

٣٢٧ الفصل الثامن : إعداد البرنامج : منهج الأجزاء ومن القمة إلى القاعدة

٨ - ٣٢ بالإشارة إلى المسألة رقم ٣٠ من هذا الفصل .. اكتب شفرة شبيهة باختبار متتالٍ sequence check يؤكد أن سجلات بطاقات العمل موجودة في ترتيب تصاعدي ؛ طبقا لرقم تعريف العامل (وعلى هذا .. يمكن أن تتفق بطريقة مناسبة مع سجلات الملف الرئيسى للعاملين ، كما هو مطلوب في المسألة رقم ٣١ من هذا الفصل) .

```
set end-of-file "off"
set previous-control-field to low-values
read input-file
  at end: set end-of-file "on"

perform until end-of-file is "on"
  if current-control-field greater than previous-control-field
    (process an in-sequence record)

    set previous-control-field to current-control-field
  else
    (current logical record is out of sequence)

    perform record-out-of-sequence-routine
  endif
  read input-file
  at end: set end-of-file "on"
endperform
```

لاحظ أن وضع قيمة ابتدائية low - value لحقل التحكم السابق .. تضمن أن حقل التحكم السابق أقل من حقل التحكم الحالى لأول سجل منطقي . لاحظ كذلك انه في كل مرة يحدث تشغيل لسجل منطقي في الترتيب ؛ فيوضع عنصر بيانات حقل التحكم السابق ، مساويا لمحتويات حقل التحكم الحالى . وهذا يعد حقل التحكم السابق للمقارنة مع السجل المنطقي التالى في الملف .

الفصل التاسع

التصحيح

Debugging

يشير التصحيح debugging إلى عملية تعريف أسباب أخطاء البرنامج وتصحيح هذه الأخطاء . وهناك ثلاثة أنواع لأخطاء البرنامج ، يجب التعامل معها ، هي :

● أخطاء تكوينية : أخطاء في القواعد عند استخدام لغة البرمجة (الكويل) ، وتكتشف بواسطة المترجم عند محاولته الترجمة من لغة الكويل الى لغة الآلة .

● أخطاء منطقية جسيمة : تتسبب في إنهاء البرنامج نهاية غير طبيعية ، وتكتشف بواسطة نظم المكونات ، أو نظام التشغيل أثناء تنفيذ البرنامج ببيانات اختبارية (أى بعد تصحيح كل الأخطاء التكوينية) ، عندما يصبح تنفيذ تعليمات لغة الآلة ، أو تصبح خدمة نظام التشغيل غير صحيحة .

● أخطاء منطقية غير جسيمة : والتي ينفذ فيها البرنامج كلية حتى عبارة STOP RUN ، ولكنه لا يعطى النتيجة الصحيحة يجب أن تزال كل الأخطاء التكوينية والأخطاء المنطقية الجسيمة من البرنامج ، قبل أن تعطى الأخطاء المنطقية غير الجسيمة فرصة لإظهار نفسها .

٩ - ١ تصحيح الأخطاء التكوينية

تعد الأخطاء التكوينية أسهل في تصحيحها ، لأن مترجم الكويل يطبع رسائل تشخيصية للخطأ ، تعرف الأخطاء ، وعبارة الكويل التي حدثت فيها هذه الأخطاء . عادة ما تظهر هذه الرسائل في نهاية قائمة برنامج المصدر ، ويفحص كل خطأ لتصحيح العبارة التي تحتويه بما يتفق مع قواعد الكويل .

مثال ٩ - ١ :

شكل ٩ - ١ ، هو قائمة برنامج مصدر بكويل IBM OS/VS ، يبين الأخطاء التكوينية مع رسائل تشخيص الخطأ . تحدد كل رسالة خطأ رقم ينتجه الكمبيوتر لبطاقة العبارة الموجود بها الخطأ ، ورمز يعرف الخطأ ودرجة خطورته ، وعبارة باللغة الإنجليزية تحاول وصف الخطأ . وفيما يلي مناقشة لأخطاء تكوينية مختارة .

```

00001 IDENTIFICATION DIVISION.
00002
00003 PROGRAM-ID. SYNTAX.
00004
00005 AUTHOR. LARRY NEWCOMER.
00006 INSTALLATION. PENN STATE UNIVERSITY -- YORK CAMPUS.
00007
00008 ENVIRONMENT DIVISION.
00009
00010 CONFIGURATION SECTION.
00011 SOURCE-COMPUTER. IBM-3081
00012 OBJECT-COMPUTER. IBM-3081.
00013
00014 INPUT-OUTPUT SECTION.
00015 FILE-CONTROL.
00016
00017     SELECT TIME-CARD-FILE
00018     ASSIGN TO TIMECARD
00019     ORGANIZATION IS SEQUENTIAL
00020     ACCESS IS SEQUENTIAL
00021
00022     SELECT VALID-TIME-FILE
00023     ASSIGN TO DISKTIME
00024     ORGANIZATION IS SEQUENTIAL
00025     ACCESS IS SEQUENTIAL
00026
00027     SELECT ERROR-LISTING-FILE
00028     ASSIGN TO ERRORLOG
00029     ORGANIZATION IS SEQUENTIAL
00030     ACCESS IS SEQUENTIAL
00031
00032
00033 DATA DIVISION.
00034
00035 FILE SECTION.
00036
00037 FD TIME-CARD-FILE
00038     RECORD CONTAINS 80 CHARACTERS
00039     LABEL RECORDS ARE OMITTED
00040
00041
00042 01 TIME-CARD-RECORD.
00043     05 TIME-CARD-ID PIC S(5).
00044     05 TIME-CARD-HOURS PIC 9(2)V9.
00045     05 TIME-CARD-X-HOURS REDEFINES TIME-CARD-HOURS
00046     PIC X(3).
00047     05 TIME-CARD-CLOSING-DATE PIC X(6).
00048     05 TIME-CARD-DEPARTMENT PIC X(4).
00049     05 FILLER PIC X(62).
00050
00051 FD VALID-TIME-FILE
00052     BLOCK CONTAINS 0 RECORDS
00053     RECORD CONTAINS 23 CHARACTERS
00054     LABEL RECORDS ARE STANDARD
00055
00056 01 VALID-TIME-RECORD.
00057     05 VALID-TIME-ID PIC X(5).
00058     05 VALID-TIME-HOURS PIC S9(2)V9 COMP-3.
00059     05 VALID-TIME-CLOSING-DATE PIC X(6).
00060     05 VALID-TIME-DEPARTMENT PIC X(4).
00061     05 VALID-TIME-EDIT-DATE PIC X(6).
00062
00063 FD ERROR-LISTING-FILE
00064     RECORD CONTAINS 132 CHARACTERS
00065     LABEL RECORDS ARE OMITTED
00066
00067 01 ERROR-REPORT-LINE PIC X(132).
00068
00069 WORKING-STORAGE SECTION.
00070
00071
00106 01 WS-HEADING-LINE-1.
00107     05 FILLER PIC X(5) VALUE SPACES.
00108     05 FILLER PIC X(15)
00109     VALUE "TIME CARD EDIT".
00110     05 WS-HEADING-MM PIC Z9.
00111     05 FILLER PIC X VALUE "//".
00112     05 WS-HEADING-DD PIC 99.
00113     05 FILLER PIC X VALUE "/".
00114     05 WS-HEADING-YY PIC 99.
00115     05 FILLER PIC X(3) VALUE SPACES.
00116     05 FILLER PIC X(5) VALUE "PAGE ".
00117     05 WS-HEADING-PAGE PIC Z99.
00118     05 FILLER PIC X(93) VALUE SPACES.

```

```

00135
00136 01 WS-UNDER-LINE.
00137 05 WS-UNDER-ID PIC X(5).
00138 05 FILLER PIC X(2) VALUE SPACES.
00139 05 WS-UNDER-HOURS PIC X(3).
00140 05 FILLER PIC X(2) VALUE SPACES.
00141 05 WS-UNDER-DATE PIC X(6).
00142 05 FILLER PIC X(2) VALUE SPACES.
00143 05 WS-UNDER-DEPARTMENT PIC X(4).
00144 05 FILLER PIC X(108) VALUE SPACES.

00158 PROCEDURE DIVISION.
00159
00160 000-EDIT-TIME-CARDS.
00161
00162 PERFORM 100-INITIALIZE
00163 * PERFORM 200-PRODUCE-EDIT-LISTING
00164 UNTIL NO-MORE-RECORDS
00165 PERFORM 300-TERMINATE
00166 STOP RUN
00167
00168 100-INITIALIZE.
00169
00170 D DISPLAY "100 ENTERED"
00171 OPEN INPUT TIME-CARD-FILE
00172 OUTPUT VALID-TIME-FILE
00173 ERROR-LISTING-FILE
00174
00175 MOVE "NO" TO WS-END-TIME-FILE-SW
00176 MOVE ZERO TO WS-PAGE-NUMBER
00177 WS-NUMBER-EMPLOYEES
00178 WS-TOTAL-HOURS
00179 WS-LINES-ON-PAGE
00180 ACCEPT WS-SYSTEM-DATE FROM DATE
00181 MOVE SYSTEM-YY TO WS-HEADING-YY

00182 MOVE SYSTEM-MM TO WS-HEADING-MM
00183 MOVE SYSTEM-DD TO WS-HEADING-DD
00184
00185 200-PRODUCE-EDIT-LISTING.
00186
00187 D DISPLAY "200 ENTERED"
00188 PERFORM 250-GET-TIME-CARD
00189 IF RECORD-AVAILABLE
00190 PERFORM 240-VALIDATE-FIELDS
00191 IF VALID-RECORD EQUAL "YES"
00192 ADD 1 TO WS-NUMBER-EMPLOYEES
00193 ADD TIME-CARD-HOURS TO WS-TOTAL-HOURS
00194 PERFORM 210-WRITE-TO-DISK
00195 ELSE
00196 PERFORM 220-PRODUCE-ERROR-LISTING
00197
00198 210-WRITE-TO-DISK.
00199
00200 D DISPLAY "210 ENTERED"
00201 MOVE TIME-CARD-ID TO VALID-TIME-ID
00202 MOVE TIME-CARD-HOURS TO VALID-TIME-HOURS
00203 MOVE TIME-CARD-CLOSING-DATE TO VALID-TIME-CLOSING-DATE
00204 MOVE TIME-CARD-DEPARTMENT TO VALID-TIME-DEPARTMENT
00205 MOVE WS-SYSTEM-DATE TO VALID-TIME-EDIT-DATE
00206 WRITE VALID-TIME-FILE
00207
00208
00209

```

CARD	ERROR MESSAGE
12	IKF10431-W END OF SENTENCE SHOULD PRECEDE OBJECT-COMPUTER . ASSUMED PRESENT.
40	IKF10561-E FILE-NAME NOT DEFINED IN A SELECT. DESCRIPTION IGNORED.
56	IKF10431-W END OF SENTENCE SHOULD PRECEDE 01 . ASSUMED PRESENT.
111	IKF21267-C VALUE CLAUSE LITERAL TOO LONG. TRUNCATED TO PICTURE SIZE.
138	IKF10171-E SPICES INVALID IN VALUE CLAUSE. SKIPPING TO NEXT CLAUSE.
172	IKF30011-E TIME-CARD-FILE NOT DEFINED. DELETING TILL LEGAL ELEMENT FOUND.
191	IKF30011-E 240 NOT DEFINED. STATEMENT DISCARDED.
191	IKF30011-E VALIDATE-FIELDS NOT DEFINED.
192	IKF40301-E FOUND " EQUAL " AFTER CONDITION. EXPECT "OR", "AND" OR VERS TO IMMEDIATELY FOLLOW
194	IKF30011-E CONDITION. DELETING TILL ONE OF THESE IS FOUND.
203	IKF30011-E TIME-CARD-HOURS NOT DEFINED. SUBSTITUTING TALLY .
204	IKF30011-E TIME-CARD-ID NOT DEFINED. DISCARDED.
205	IKF30011-E TIME-CARD-HOURS NOT DEFINED. DISCARDED.
206	IKF30011-E TIME-CARD-CLOSING-DATE NOT DEFINED. DISCARDED.
206	IKF30011-E TIME-CARD-DEPARTMENT NOT DEFINED. DISCARDED.

• **البطاقة 12 :** فى كويل IBM OS / VS ، رمز الرسالة الذى ينتهى بالحرف W (مثل W - 10431 IKF) يحدد تحذيرا Warning ، وهذا يعنى أن الخطأ بسيط ، وأن المترجم كان قادرا على تصحيحه . فى هذه الحالة ، نسى المبرمج النقطة التى تنهى مقطع كمبيوتر المصدر ، وقد افترض المترجم أن النقطة موجودة ، على أية حال .. يستطيع المترجم ترجمة العبارات التى تنتج رسائل تحذيرية .

• **البطاقة 111 :** فى كويل IMB OS/VS ، يحدد رمز الرسالة الذى ينتهى بالحرف C (مثل C - 2126I IKF) أخطاء شرطية conditional errors . هذه الأخطاء أكثر شدة من الأخطاء التحذيرية ، إلا أن المترجم لا يزال قادرا على تخمين ما يريده المبرمج . يستخدم هذا التخمين فى ترجمة العبارة ، إلا أن احتمال أن الترجمة هى ما يعنيه المبرمج ليس مرتفعاً كما فى حالة الأخطاء التحذيرية . فى هذه الحالة ، كان جزء VALUE أكبر من حجم PICTURE ، وعلى هذا .. حذف المترجم مازاد عن عدد الخانات ("/" VALUE أصبحت "/" لتتاسب PIC X) .

• **البطاقة 192 :** فى كويل IBM OS / VS ، رمز الرسالة الذى ينتهى بالحرف E (مثل E - 4030I IKF) يحدد أخطاء جسيمة severe errors . فى هذه الحالة .. لا يستطيع المترجم حتى أن يخمن ما تعنيه العبارة . وعلى هذا ، لا يمكن ترجمة العبارة إلى لغة الآلة .. ولذلك .. لن يتواجد برنامج هدف لتنفيذ بيانات اختبارية . لاحظ أنه فى هذه اللحظة لا تذكر رسالة الخطأ الإنجليزية للمبرمج الخطأ الموجود فعلا فى العبارة . والخطأ الفعلى هو أن المبرمج استخدم اسم شرطى - VALID RECORD بدلا من اسم بيانات SW - ERROR - DETECTION - WS فى شرط العلاقة VALID- RECORD EQUAL "YES"

• **البطاقة 40 :** هذا الخطأ التكويني يوضح حقيقة أن رسائل الخطأ التشخيصية لا تشير دائما إلى رقم سطر العبارة التى تسببت فعلا فى الخطأ . وهنا .. يعد السطر 40 نقطة مرتبة تنهى FILE - CARD - TIME . وتقول رسالة الخطأ NAME NOT DEFINED IN A` SELECT . بالرغم من أن رسالة الخطأ تشير للسطر 40 .. إلا أن المشكلة الفعلية هى فى خطأ كتابة فى السطر 17 ، والذى يقرأ FILE - CARD - TIME - SELECT .

• **البطاقات 194, 203, 204, 205, 206 :** بسبب خطأ البطاقة 40 .. لم يتم تشغيل FD ووصف المستوى 01 للملف - TIME CARD - FILE بواسطة المترجم . وعلى هذا .. تشتمل كل إشارات جزء الإجراءات إلى الحقول الموجودة فى - TIME CARD - RECORD على أخطاء منطقية . وتقول رسالة الخطأ إن الأوقات غير معرفة NOT DEFINED .. ومن وجهة نظر المترجم .. فهى بالرغم من تعريف المبرمج لها فى الأسطر من 42 إلى 49 فى جزء البيانات غير معرفة للمترجم .

٩ - ٢ تصحيح الأخطاء المنطقية الجسيمة

عندما يتسبب خطأ جسيم فى إنهاء البرنامج نهاية غير طبيعية .. يلغى نظام التشغيل تنفيذ بقية البرنامج ، بل أكثر من ذلك أنه إذا قامت لغة تحكم البرنامج الخاصة بالبرنامج بطليه .. فإن نظام التشغيل يطبع نفايا الفصل termination dump (فى ملف طباعة محدد فى لغة تحكم العمل) . تقدم نفايا الفصل أحد أو كلا نوعى المعلومات التالية والمفيدة فى تصحيح الأخطاء المنطقية الجسيمة :

نفايا ذاكرة : توضح نفايا الذاكرة محتويات ذاكرة الكمبيوتر فى وقت حدوث الخطأ الجسيم ، ويمكن عادة تحديد سبب الخطأ بدراسة محتويات عناصر بيانات حرجة .

تتبع برنامج : يسجل تتبع البرنامج program trace كل التعليمات المنفذة بواسطة الكمبيوتر قبل وعند وقت إنهاء البرنامج نهاية غير طبيعية . فمثلا .. يمكن أن يطبع رقم السطر الذى ينتجه المترجم لكل عبارة قبل ترجمة لغة الآلة مباشرة للعبارة التى نفذت .

فى بعض نظم الكمبيوتر ، لا تتاح إلا نفايا سادسة عشرية نمطية لنظام التشغيل فقط ، وهذه هى نفايا الذاكرة مطبوعة ، كسلسلة من أرقام للأساس 16 . ويجب أن يتعلم مبرمج الكويل الجاد لغة مجمع assembler language على الأقل ليكون قادرا على تفسير نفايا ذاكرة الكمبيوتر . تقدم بعض أجهزة الكمبيوتر الأخرى نفايا رمزية symbolic dumps حيث تطبع محتويات بعض أو كل عناصر جزء البيانات ، عندما ينتهى البرنامج نهاية غير طبيعية (نفايا ذاكرة) ، وذلك مع تتبع البرنامج ، وتطبع عناصر البيانات فى حالة DISPLAY مع تسميتها بأسماء كويل .

٩ - ٣ تصحيح الأخطاء المنطقية غير الجسيمة

بينما تنتج الأخطاء التكوينية رسائل ، وتنتج الأخطاء المنطقية الجسيمة نفايا الإنهاء .. فإن الأخطاء المنطقية غير الجسيمة تكون أكثر صعوبة فى التصحيح ، حيث لا تتحدد عبارة خاصة بأنها سبب الخطأ . ويترك تحديد الخطأ وسببه لبراعة المبرمج .

هناك بعض سمات للغة الكويل ، صممت خصيصا لمساعدة المبرمج فى تعريف الأخطاء المنطقية غير الجسيمة . ونظرا لأن هذه الطرق والعبارات الخاصة مفيدة جدا كذلك فى توفير معلومات ، عن حالة تنفيذ البرنامج قبل حدوث خطأ منطقى جسيم (انظر مثال ٩ - ٦) فهى مقدمة فى الأقسام من رقم 4 إلى رقم 6 من هذا الفصل .

٩ - ٤ الحصول على معلومات تتبع البرنامج

إحدى طرق الحصول على معلومات تتبع البرنامج ، هى وضع سطر تصحيح (مع وضع D فى عموده السابع) ، فى بداية كل مقطع من مقاطع جزء الإجراءات ، يحتوى هذا السطر على عبارة DISPLAY . وكلما تم إدخال مقطع .. تنفذ عبارة DISPLAY لتطبع رسالة تخدم كإشارة تتبع للبرنامج .

مثال ٩ - ٢ :

SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE.

.....
PROCEDURE DIVISION.

D DISPLAY "PROGRAM EXECUTION BEGINNING"

.....
PARA-ONE.

D DISPLAY "PARA-ONE BEING ENTERED"

.....
PARA-TWO.

D DISPLAY "PARA-TWO BEING ENTERED"

.....

عندما يتم تصحيح البرنامج .. يمكن حذف WITH DEBUGGING MODE من قطع كمبيوتر المصدر . وتعامل كل الأسطر الموجودة D في عمودها السابع عند ذلك كتعليقات ، وهذا يحذف مخرجات التتبع . ويقدم كويل IBM OS / VS عبارة خاصة ، تهدف كذلك إلى إنتاج تتبع البرنامج .

READY TRACE

والتي تلغى عمل التتبع . يمكن تنفيذ التتبع مع READY TRACE ، وإعادة إعداد التتبع RESET TRACE ، لاي عدد مطلوب من المرات في البرنامج .

مثال ٩ - ٢ :

يستخدم جزء الإجراءات في مثال ٩ - ٢ عبارة READY TRACE ، وموضع عينة للمخرجات في نهايته . وعبارات /READY RESET TRACE في كل المخرجات .. ففي كويل IBM OS / VS تطبع READY TRACE مخرجاتها على ملف نظام خاص SYSOUT . وعلى هذا ، يجب أن توجد عبارة لغة تحكم العمل لهذا الملف ، عندما تستخدم READY TRACE (وإلا فسوف ينتهي البرنامج نهاية غير طبيعية) . لاحظ أن هذا التتبع يطبع اسم مقطع الكويل ، أول عبارة في المقطع المنفذ مباشرة . وتحت لغات تحكم عمل مختلفة .. يمكن أن يطبع رقم السطر الذي ينتجه المترجم بدلا من ذلك .

```

00070      PROCEDURE DIVISION.
00072      *-----
00073      READY TRACE
00074      *-----
00075
00076      MOVE "NO " TO END-OF-CARDS-SWITCH
00077      MOVE ZERO TO TOTAL-SALES
00078
00079      OPEN      INPUT      CARD-FILE
00080              OUTPUT      PRINT-FILE
00081
00082      PERFORM 100-INPUT-A-RECORD
00083
00084      PERFORM 200-PRODUCESALES-LISTING
00085              UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00086
00087      WRITE PRINT-LINE
00088              FROM SALES-TOTAL-LINE
00089
00090      CLOSE      CARD-FILE
00091              PRINT-FILE
00092      STOP RUN
00093
00095      100-INPUT-A-RECORD.
00096
00097      READ CARD-FILE RECORD
00098      INTO WORKING-SALES-CARD
00099      AT END
00100      MOVE "YES" TO END-OF-CARDS-SWITCH
00101
00103      200-PRODUCESALES-LISTING.
00104
00105      ADD WORKING-SALES-AMOUNT TO TOTAL-SALES
00106
00107      MOVE WORKING-SALES-ID      TO LISTING-SALES-ID
00108      MOVE WORKING-CUSTOMER-ID  TO LISTING-CUSTOMER-ID
00109      MOVE WORKING-SALES-AMOUNT TO LISTING-SALES-AMOUNT
00110
00111      WRITE PRINT-LINE
00112              FROM SALES-LISTING-LINE
00113
00114      PERFORM 100-INPUT-A-RECORD
00115

```

شكل (٩ - ٢)

100-INPUT-A-RECORD , 200-PRODUCESALES-LISTING , 100-INPUT-A-RECORD , 200-PRODUCESALES-LISTING ,
 100-INPUT-A-RECORD , 200-PRODUCESALES-LISTING , 100-INPUT-A-RECORD , 200-PRODUCESALES-LISTING ,
 100-INPUT-A-RECORD , 200-PRODUCESALES-LISTING , 100-INPUT-A-RECORD , 200-PRODUCESALES-LISTING ,
 100-INPUT-A-RECORD ,

شكل (٩ - ٢) تكملة

وهناك ملحقات قوى لعبارات READY/RESET TRACE ميسر عن طريق التوسع في كويل IBM OS/VS عن IMB النمطى ANS التالى :

ON {integer-1} [AND EVERY {integer-2}] [UNTIL {integer-3}]
 {identifier-1} {identifier-2} {identifier-3}
 imperative statement
 [ELSE]
 imperative statement

مثال ٩ - ٤ :

SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE:

PROCEDURE DIVISION.

PERFORM PARA-A
 UNTIL SOME-CONDITION-OR-OTHER

PARA-A.

D ON 1 AND EVERY 2 UNTIL 9
 D READY TRACE
 D ELSE
 D RESET TRACE
 D

تنفذ READY TRACE فى أول مرة ، وثالث مرة ، وخامس مرة ، وسابع مرة ، وتساع مرة يدخل فيها المقطع PARA-A وتنفذ RESET TRACE (بدلاً من تنفيذ READY TRACE) لكل مرات التنفيذ الأخرى للمقطع . لاحظ استخدام D فى العمود رقم 7 فى كل عبارات التصحيح (بما فى ذلك النقطة المرتبة فى نهاية عبارة ON) . وعلى هذا .. تترجم هذه العبارات بواسطة المترجم ، عندما يتحدد WITH DEBUGGING MODE فقط فى مقطع كمبيوتر المصدر .

وفيما يلى أمثلة أكثر لعبارات ON صحيحة :

- ON 3 AND EVERY 5 READY TRACE ELSE RESET TRACE.
- ON 1 READY TRACE ELSE RESET TRACE.
- ON 1 AND EVERY 2 DISPLAY INPUT-RECORD:
- ON 1 AND EVERY 2
 DISPLAY INPUT-RECORD
 READY TRACE
 ELSE
 RESET TRACE

٩ - ٥ اخراج عناصر بيانات أثناء تنفيذ البرنامج

أبسط طريقة للحصول على معلومات نفايا ذاكرة ديناميكية dynamic memory dump هي إدخال أسطر تصحيح بعبارات DISPLAY في البرنامج . وعندما تنفذ عبارة DISPLAY .. فإنها تطبع محتويات عنصر البيانات المحدد . وبعد عرض DISPLAY كل سجل مدخلات بعد عبارة READ ، وكل سجل مخرجات قبل عبارة WRITE هو فكرة طيبة .

مثال ٩ - ٥ :

```
SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE.
.....
PROCEDURE DIVISION.
.....
    READ INPUT-FILE
    AT END MOVE "YES" TO END-FILE-SW

D    DISPLAY "INPUT RECORD IS:" INPUT-REC
.....
D    DISPLAY "OUTPUT RECORD IS:" OUTPUT-REC
    WRITE OUTPUT-REC
.....
    ADD 1 TO SAMPLE-COUNTER
D    DISPLAY "SAMPLE-COUNTER AFTER INCREMENTING:" SAMPLE-COUNTER
.....
```

لاحظ استخدام الثوابت غير العددية في عبارات DISPLAY في تسمية مخرجات التصحيح . وتوفر هذه الأسماء معلومات تتبع برنامج كذلك ؛ حيث إنها تحدد موقع تنفيذ الكمبيوتر لتعليمات البرنامج ، عندما تنتج مخرجات DISPLAY . ويمكن لمخرجات العبارات السابقة أن تشبه ما يلي :

```
INPUT RECORD IS:0003A789108264
OUTPUT RECORD IS:172A12 CARTONSB97634900000
SAMPLE-COUNTER AFTER INCREMENTING:0007
```

تذكر : عند استخدام DISPLAY في كويل IBM OS/VS : يجب تقديم عبارة لغة تحكم عمل لتعريف ملف الطابع SYSOUT ، وإلا فسوف ينتهي البرنامج نهاية غير طيبيعية .

يشمل عديد من صيغ الكويل عبارة لا تطبع محتويات عنصر البيانات فقط ، بل تطبع كذلك اسم الكويل لعنصر البيانات ، والتكوين كما يلي :

$$\text{EXHIBIT} \left\{ \begin{array}{l} \text{NAMED} \\ \text{CHANGED NAMED} \\ \text{CHANGED} \end{array} \right\} \left\{ \begin{array}{l} \text{identifier-1} \\ \text{literal-1} \end{array} \right\} \left[\begin{array}{l} \text{identifier-2} \\ \text{literal-2} \end{array} \right] \dots$$

يطبع EXHIBIT NAMED محتويات كل عنصر بيانات مع اسم الكويل للعنصر ، في الصورة : con-cobol-name = tents

يطبع EXHIBIT CHANGED NAMED اسم الكويل للعنصر تتبعه محتويات الحالية ، بشرط أن المحتويات منذ آخر مرة نفذت فيها عبارة EXHIBIT الخاصة هذه .

يطبع EXHIBIT CHANGED محتويات عنصر البيانات المحدد ، بشرط أن تتغير منذ آخر مرة ، نفذت فيها عبارة EXHIBIT هذه ، إذا لم يطبع اسم الكويل .

وفي كويل IBM OS/VS .. يجب تقديم عبارة لغة تحكم العمل ؛ لتعريف ملف طباعة SYSOUT عند استخدام أى صورة من صور EXHIBIT .

مثال ٩ - ٦ :

أضيفت عبارات EXHIBIT NAMED, READY TRACE الى برنامج شكل ٩ - ٢ ؛ للمساعدة في تصحيح إنهاء البرنامج نهاية غير طبيعية . توضح مخرجات التصحيح ، شكل (٩ - ٤) كيف خلطت مخرجات EXHIBIT, READY TRACE مع بعضها البعض في ملف طباعة SYSOUT . وكيف تبدأ EXHIBIT سطورا جديدا تلقائيا ، إذا لم يكف السطر الواحد كل المعلومات ، وكيف قدمت مخرجات READY TRACE تاريخا مفيدا لما حدث أثناء تنفيذ البرنامج . قارن التقرير المطبوع في نهاية شكل ٩ - ٢ مع مخرجات التصحيح ؛ للتأكد من أنك تتبع منطق البرنامج مع تشغيله لكل سجل مدخلات . تم تنفيذ آخر مقطع (طبقا للتتبع) هو PRODUCE - TIME - LISTING - 200 . لاحظ أن الكمبيوتر ذهب على الأقل - إلى نفس بعد عبارة EXHIBIT في هذا المقطع ، حيث تظهر محتويات WORKING - OVERTIME - HOURS WORKING - REGULAR- HOURS في مخرجات التصحيح . نستطيع أن نستخلص أن النهاية غير الطبيعية حدثت أثناء تنفيذ عبارة COMPUTE في السطر 116 ؛ لان المخرجات من عبارة EXHIBIT توضح أن WORKING - OVER- TIME - HOURS يحتوى على فراغات . ولم تكن هذه النهاية غير الطبيعية كذلك بسبب البرنامج ، بل كانت بسبب بيانات مدخلات غير صحيحة ويجب إعادة تصميم البرنامج بحيث تتفع (يتأكد من صحة) بيانات المدخلات قبل محاولة تشغيلها .

```

00001      IDENTIFICATION DIVISION.
00002
00003      PROGRAM-ID. TIMELIST.
00004
00005      AUTHOR. LARRY NEWCOMER.
00006      INSTALLATION. PENN STATE UNIVERSITY, YORK CAMPUS.
00007
00008      DATE-COMPILED. MAY 9,1983.

00011      ENVIRONMENT DIVISION.

00013      CONFIGURATION SECTION.

00015      SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE.
00016      OBJECT-COMPUTER. IBM-3081.

00018      INPUT-OUTPUT SECTION.

00020      FILE-CONTROL.

00022          SELECT CARD-FILE          ASSIGN TO CARDS.
00023          SELECT PRINT-FILE         ASSIGN TO PRINTER.

00025      DATA DIVISION.

00027      FILE SECTION.

00029      FD CARD-FILE
00030          RECORD CONTAINS 80 CHARACTERS
00031          LABEL RECORDS ARE OMITTED
00032          .
00033
00034      01 TIME-CARD-INPUT          PIC X(80).

00036      FD PRINT-FILE
00037          RECORD CONTAINS 132 CHARACTERS
00038          LABEL RECORDS ARE OMITTED
00039          .
00040
00041      01 PRINT-LINE          PIC X(132).

00043      WORKING-STORAGE SECTION.

00045      01 END-OF-CARDS-SWITCH          PIC X(3).

00047      01 WORKING-TIME-CARD.

00048
00049          05 WORKING-NAME          PIC X(20).
00050          05 WORKING-REGULAR-HOURS  PIC 99.
00051          05 WORKING-OVERTIME-HOURS PIC 99.
00052          05 FILLER                PIC X(56).

00054      01 TIME-LISTING-LINE.

00055
00056          05 LISTING-NAME          PIC X(20).
00057          05 FILLER                PIC X(5)      VALUE SPACES.
00058          05 LISTING-REGULAR-HOURS PIC 99.
00059          05 FILLER                PIC X(5)      VALUE SPACES.
00060          05 LISTING-OVERTIME-HOURS PIC 99.
00061          05 FILLER                PIC X(5)      VALUE SPACES.

```

شكل (٩ - ٣)

```

00062      05 LISTING-TOTAL-HOURS      PIC 999.
00063      05 FILLER                     PIC X(85)      VALUE SPACES.

00065      01 EMPLOYEE-TOTAL-LINE.
00066
00067      05 FILLER                       PIC X(25)
00068      VALUE "NUMBER OF EMPLOYEES IS".
00069      05 NUMBER-OF-EMPLOYEES         PIC 999.
00070      05 FILLER                     PIC X(104)     VALUE SPACES.

00072      PROCEDURE DIVISION.

00074      D    READY TRACE
00075
00076      MOVE "NO " TO END-OF-CARDS-SWITCH
00077      MOVE ZERO  TO NUMBER-OF-EMPLOYEES
00078
00079      OPEN      INPUT      CARD-FILE
00080              OUTPUT     PRINT-FILE
00081
00082      PERFORM 100-INPUT-A-RECORD
00083
00084      PERFORM 200-PRODUCE-TIME-LISTING
00085              UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00086
00087      WRITE PRINT-LINE
00088              FROM EMPLOYEE-TOTAL-LINE
00089
00090      CLOSE     CARD-FILE
00091              PRINT-FILE
00092      STOP RUN
00093      .

00095      100-INPUT-A-RECORD.
00096
00097      READ CARD-FILE RECORD
00098          INTO WORKING-TIME-CARD
00099          AT END
00100              MOVE "YES" TO END-OF-CARDS-SWITCH
00101      .
00102      D,    EXHIBIT NAMED WORKING-TIME-CARD
00103      D      END-OF-CARDS-SWITCH
00104      D      .

00106      200-PRODUCE-TIME-LISTING.
00107
00108      ADD 1 TO NUMBER-OF-EMPLOYEES
00109
00110      MOVE WORKING-NAME      TO LISTING-NAME

00111      MOVE WORKING-REGULAR-HOURS TO LISTING-REGULAR-HOURS
00112      MOVE WORKING-OVERTIME-HOURS TO LISTING-OVERTIME-HOURS
00113
00114      D    EXHIBIT NAMED WORKING-REGULAR-HOURS
00115      D      WORKING-OVERTIME-HOURS
00116      COMPUTE LISTING-TOTAL-HOURS = WORKING-REGULAR-HOURS
00117                                      + WORKING-OVERTIME-HOURS
00118
00119      WRITE PRINT-LINE
00120              FROM TIME-LISTING-LINE

```

شكل (٩ - ٣) تكملة

00121
00122 PERFORM 100-INPUT-A-RECORD
00123

PRINTED REPORT:

ABEL FRED	40	03	043
BAKER SUE	30	00	030
CHARLIE CHUCK	40	12	052
PERELMAN BARNEY	40	15	055
PERELMAN MIKE	03	00	003

شكل (٩ - ٣) تكملة

100-INPUT-A-RECORD , WORKING-TIME-CARD = ABEL FRED = NO	4003	END-OF-CARDS-SWITCH
200-PRODUCE-TIME-LISTING , WORKING-REGULAR-HOURS = 40 WORKING-OVERTIME-HOURS = 03		
100-INPUT-A-RECORD , WORKING-TIME-CARD = BAKER SUE = NO	3000	END-OF-CARDS-SWITCH
200-PRODUCE-TIME-LISTING , WORKING-REGULAR-HOURS = 30 WORKING-OVERTIME-HOURS = 00		
100-INPUT-A-RECORD , WORKING-TIME-CARD = CHARLIE CHUCK = NO	4012	END-OF-CARDS-SWITCH
200-PRODUCE-TIME-LISTING , WORKING-REGULAR-HOURS = 40 WORKING-OVERTIME-HOURS = 12		
100-INPUT-A-RECORD , WORKING-TIME-CARD = PERELMAN BARNEY = NO	4015	END-OF-CARDS-SWITCH
200-PRODUCE-TIME-LISTING , WORKING-REGULAR-HOURS = 40 WORKING-OVERTIME-HOURS = 15		
100-INPUT-A-RECORD , WORKING-TIME-CARD = PERELMAN MIKE = NO	0300	END-OF-CARDS-SWITCH
200-PRODUCE-TIME-LISTING , WORKING-REGULAR-HOURS = 03 WORKING-OVERTIME-HOURS = 00		
100-INPUT-A-RECORD , WORKING-TIME-CARD = TROUBLE TOM = NO	40	END-OF-CARDS-SWITCH
200-PRODUCE-TIME-LISTING , WORKING-REGULAR-HOURS = 40 WORKING-OVERTIME-HOURS =		

شكل (٩ - ٤)

٩ - ٦ التوضيحات وعبارة يستخدم للتصحيح

يقدم الكويل النمطى ANS توضيحات DECLARATIVES لأجزاء تصحيح خاصة ، ويجب أن تكتب هذه التوضيحات في بداية جزء الإجراءات . ومثل هذه الأجزاء فريدة ، في أنها لا تنفذ كجزء من منطق البرنامج المعتاد ، وإنما تنفذ إذا ما أثبتت

بواسطة أحداث خاصة محددة فقط .

مثال ٩ - ٧ :

```
PROCEDURE DIVISION.
DECLARATIVES.
DECLARATIVE-SECTION-1 SECTION.
.....
DECLARATIVE-SECTION-2 SECTION.
.....
END DECLARATIVES.
... (normal PROCEDURE DIVISION statements)
```

... (عبارة عادية من عبارات جزء الإجراءات)

لاحظ العنوان DECLARATIVES ، وآخر سطر END DECLARATIVES (المكتوبة في المنطقة A) .

يجب أن يبدأ كل قسم توضيحات DECLARATIVE SECTION بعبارة USE ، والتي تحدد متى ينفذ قسم التوضيحات . يعرض الشكل العام لتوضيحات التصحيح DEBUGGING DECLARATIVES في شكل ٩ - ٥ .

USE FOR DEBUGGING

ON	<div style="display: inline-block; vertical-align: middle;"> <div style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 5px;"> [ALL REFERENCES OF] identifier-1 file-name-1 procedure-name-1 ALL PROCEDURES </div> </div>	<div style="display: inline-block; vertical-align: middle;"> <div style="border-left: 1px solid black; border-right: 1px solid black; padding: 0 5px;"> [ALL REFERENCES OF] identifier-2 file-name-2 procedure-name-2 ALL PROCEDURES </div> </div>	...
----	---	---	-----

شكل (٩ - ٥)

مثال ٩ - ٨ :

```
PROCEDURE DIVISION.
DECLARATIVES.
CHECK-COUNTER SECTION.
  USE FOR DEBUGGING
    ON ALL REFERENCES OF NUMBER-CUSTOMERS-BILLED
      EXHIBIT NAMED NUMBER-CUSTOMERS-BILLED
      EXHIBIT NAMED CURRENT-CUSTOMER-ID
END DECLARATIVES.
```

استخدم هنا جزء identifier-1 : ON ALL REFERENCE OF . وينفذ القسم (في هذه الحالة .. عبارتي EXHIBIT NAMED) قبل كل كتابة WRITE معرف identifier-1 مباشرة ، وبعد أي عبارة كويل تحتوي - صراحة ومباشرة - علي identifier-1 .

وفيما يلي بدائل أخرى لـ USE FOR DEBUGGING .

● ON identifier-1 (ON SAMPLE-OUTPUT-RECORD)

عادة ما يكون identifier-1 اسم سجل كويل للملف . مخرجات تنفذ التوضيحات DECLARATIVE قبل كل عبارة WRITE في الملف مباشرة (مثل قبل عبارة WRITE SAMPLE - OUTPUT - RECORD) .

● ON file-name-1 (ON EMPLOYEE-MASTER-FILE)

تنفذ التوضيحات DECLARATIVE بعد أي عبارة OPEN أو عبارة CLOSE ، أو عبارة READ للملف المحدد .

● ON procedure-name-1 (ON 200-PRODUCE-MAILING-LABELS)

procedure - name - 1 هو اسم مقطع أو اسم قسم في جزء الإجراءات ، ينفذ التوضيحات قبل كل تنفيذ للإجراء المسمى .

● ON ALL PROCEDURES

تنفذ التوضيحات قبل كل تنفيذ لكل المقاطع وكل الأقسام (باستثناء أقسام التوضيحات نفسها) .
وكما هو الحال مع أسطر التصحيح الأخرى .. تترجم توضيحات التصحيح DEBUGGING DECLARATIVES إلى برنامج هدف بلغة الآلة إذا حدد WITH DEBUGGING MODE في مقطع برنامج المصدر فقط .
عندما تنفذ توضيحات التصحيح .. يضع النظام معلومات قيمة في منطقة بيانات خاصة ، تسمى DEBUG - ITEM .
وكلمة DEBUG - ITEM هي كلمة كويل محجوزة ، لا يعرفها المبرمج في جزء البيانات ، وهي لها تخطيط الكويل المحدد في شكل ٩ - ٦ .

01 DEBUG-ITEM.

05 DEBUG-LINE	PIC X(6).	
05 FILLER	PIC X	VALUE SPACES.
05 DEBUG-NAME	PIC X(30).	
05 FILLER	PIC X	VALUE SPACES.
05 DEBUG-SUB-1	PIC \$9999	SIGN IS LEADING SEPARATE CHARACTER.
05 FILLER	PIC X	VALUE SPACES.
05 DEBUG-SUB-2	PIC \$9999	SIGN IS LEADING SEPARATE CHARACTER.
05 FILLER	PIC X	VALUE SPACES.
05 DEBUG-SUB-3	PIC \$9999	SIGN IS LEADING SEPARATE CHARACTER.
05 FILLER	PIC X	VALUE SPACES.
05 DEBUG-CONTENTS	PIC X(n).	

شكل (٩ - ٦)

تكون محتويات حقول المستوى 05 كما يلي :

- ١ - يوضع DEBUG - LINE لرقم سطر ينتجه المترجم للعبارة التي تسببت في تنفيذ DECLARATIVE .
- ٢ - يوضع DEBUG - NAME لأول 30 رمز من الاسم الذي تسبب في تنفيذ DECLARATIVES .
- ٣ - يوضع 1 - SUB - DEBUG , 2 - SUB - DEBUG , 3 - SUB - DEBUG لقيم الدليل الحالية للعنصر -DEBUG - NAME , إذا كان جزءا من جدول (انظر الفصل العاشر) ، وإلا.. فسوف تترك هذه الأجزاء من DEBUG - ITEM فارغة.
- ٤ - توضع DEBUG - CONTENTS لمحتويات عنصر البيانات (الذي يكون اسمه في DEBUG - NAME) الذي تسبب في تنفيذ DECLARATIVES فإذا كان DECLARATIVE قد نفذ بسبب USE FOR DEBUGGING ON Proc- dure. name فإن DEBUG - NAME تحتوى على اسم الإجراءات ، ويوضع DEBUG - CONTENTS لعبارة إنجليزية تصف نوع الإجراء المعنى .
- يحتوى DEBUG - ITEM على معلومات مفيدة جدا بحيث يكون استخدام EXHIB- IT DEBUG -ITEM بسيطة هو كل المطلوب في قسم التوضيحات .

مثال ٩ - ٩ :

نعيد اعتبار النهاية غير الطبيعية في مثال ٩ - ٦ ، مع استخدام التوضيحات كوسيلة تصحيح في هذه المرة . جزء الإجراءات الجديد موضح في شكل ٩ - ٧ ، مع تحديد الأسطر من ٧٤ الى ٨٨ للتوضيحات . عندما يحدد عديد من الشروط في جزء ON واحد ، يتسبب حدوث أى منها في تنفيذ التوضيحات . لاحظ أن كل التوضيحات تكون - EXHIBIT DEBUG - ITEM . تبين عينة المخرجات ، شكل ٩ - ٨ ، كم المعلومات الذي تظهره هذه العبارة البسيطة . (هيئة شكل ٩ - ٨ تتبع شكل ٩ - ٦ ، مع فراغات في SUB - DEBUG ومع PERFORM LOOP بانها DEBUG - CONTENTS 200 - PRODUCE - LISTING - TIME) .. نستدل أن البرنامج ذهب على الأقل حتى السطر ١٢٤ (يظهر آخر سطر في مخرجات التصحيح ، ويحتوى WORKING - OVERTIME - HOURS عند هذه النقطة ، على فراغات . وحيث إن السطر ١٢٥ في برنامج الكويل فارغ ويستخدم السطر ١٢٦ WORKING - OVERTIME - HOURS (المحتوى على فراغات) في حسابات .. تتضح مشكلة الإنهاء غير الطبيعي (انظر مثال ٩ - ٦) .

```

00072      PROCEDURE DIVISION.
00074      DECLARATIVES.
00075
00076      IDENTIFY-ABEND SECTION.
00077
00078          USE FOR DEBUGGING
00079              ON ALL REFERENCES OF WORKING-TIME-CARD
00080                  ALL REFERENCES OF WORKING-REGULAR-HOURS
00081                  ALL REFERENCES OF WORKING-OVERTIME-HOURS
00082                  ALL REFERENCES OF 200-PRODUCE-TIME-LISTING
00083
00084

```

شكل (٩ - ٧)

```

00085      EXHIBIT NAMED DEBUG-ITEM
00086      .
00087
00088      END DECLARATIVES.
00089
00090
00091      MOVE "NO " TO END-OF-CARDS-SWITCH
00092      MOVE ZERO TO NUMBER-OF-EMPLOYEES
00093
00094      OPEN      INPUT      CARD-FILE
00095              OUTPUT      PRINT-FILE
00096
00097      PERFORM 100-INPUT-A-RECORD
00098
00099      PERFORM 200-PRODUCE-TIME-LISTING
00100              UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00101
00102      WRITE PRINT-LINE
00103              FROM EMPLOYEE-TOTAL-LINE
00104
00105      CLOSE CARD-FILE
00106              PRINT-FILE
00107      STOP RUN
00108      .

00110      100-INPUT-A-RECORD.
00111
00112      READ CARD-FILE RECORD
00113              INTO WORKING-TIME-CARD
00114              AT END
00115              MOVE "YES" TO END-OF-CARDS-SWITCH
00116

00118      200-PRODUCE-TIME-LISTING.
00119
00120      ADD 1 TO NUMBER-OF-EMPLOYEES
00121
00122      MOVE WORKING-NAME      TO LISTING-NAME
00123      MOVE WORKING-REGULAR-HOURS TO LISTING-REGULAR-HOURS
00124      MOVE WORKING-OVERTIME-HOURS TO LISTING-OVERTIME-HOURS
00125
00126      COMPUTE LISTING-TOTAL-HOURS = WORKING-REGULAR-HOURS
00127                                  + WORKING-OVERTIME-HOURS
00128
00129      WRITE PRINT-LINE
00130              FROM TIME-LISTING-LINE
00131
00132      PERFORM 100-INPUT-A-RECORD
00133

```

شكل (٧-٩) بكلمة

DEBUG-ITEM = 000112 WORKING-TIME-CARD	ABEL FRED	4003
DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING	PERFORM LOOP	
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS	40	
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS	03	
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS	40	
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS	03	
DEBUG-ITEM = 000112 WORKING-TIME-CARD	BAKER SUE	3000
DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING	PERFORM LOOP	
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS	30	
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS	00	
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS	30	
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS	00	
DEBUG-ITEM = 000112 WORKING-TIME-CARD	CHARLIE CHUCK	4012
DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING	PERFORM LOOP	
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS	40	
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS	12	
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS	40	
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS	12	
DEBUG-ITEM = 000112 WORKING-TIME-CARD	PERELMAN BARNEY	4015
DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING	PERFORM LOOP	
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS	40	
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS	15	
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS	40	
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS	15	
DEBUG-ITEM = 000112 WORKING-TIME-CARD	PERELMAN MIKE	0300
DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING	PERFORM LOOP	
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS	03	
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS	00	
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS	03	
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS	00	
DEBUG-ITEM = 000112 WORKING-TIME-CARD	TROUBLE TOM	40
DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING	PERFORM LOOP	
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS	40	
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS		

شكل (٩ - ٨)

اسئلة مراجعة

- ٩ - ١ ماهو اختبار البرنامج ؟
- ٩ - ٢ ناقش ثلاثة أنواع من الأخطاء التي يمكن أن تحدث أثناء اختبار البرنامج .
- ٩ - ٣ ماهي رسائل الخطأ التشخيصية ؟
- ٩ - ٤ ناقش الاختلافات بين الأخطاء التكوينية من المستوى W والمستوى E .
- ٩ - ٥ ماهي نفايا الذاكرة ؟ ميز بين النفايا السادسة عشرية والنفايا الرمزية .
- ٩ - ٦ ماهو تتبع البرنامج ؟ كيف يكون مفيداً في التصحيح ؟

- ٩ - ٧ ما هي المعلومات التي تقدم بصورة تقليدية في النفايا الرمزية ؟
- ٩ - ٨ كيف يمكن استخدام عبارات DISPLAY في التصحيح ؟
- ٩ - ٩ صف آثار RESET TRACE, READY TRACE
- ٩ - ١٠ صف تكوين واستخدام عبارة ON
- ٩ - ١١ ناقش تكوين واستخدام عبارة EXHIBIT
- ٩ - ١٢ ما شكل المخرجات من EXHIBIT NAMED
- ٩ - ١٣ ما هي التوضيحات DECLARATIVES ؟
- ٩ - ١٤ كيف تختلف التوضيحات عن العبارات الأخرى في جزء الإجراءات ؟
- ٩ - ١٥ ناقش تأثير مقطع كمبيوتر المصدر على التوضيحات ، والأسطر التي بها D في عمودها السابع
- ٩ - ١٦ ناقش تأثيرات ما يلي في عبارة USE FOR DEBUGGING

ON ALL REFERENCES (١)

ON file - name (ب)

ON procedure - name (ج)

ON ALL PROCEDURES (د)

- ٩ - ١٧ ناقش محتويات DEBUG - ITEM ومدى فائدتها

مسائل محلولة

- ٩ - ١٨ صحح البرنامج التالي ، الذي يحتوى على أخطاء منطقية غير جسيمة . معطى لك قائمة ببرنامج المصدر (شكل ٩ - ١)
- ٩ - ١٩ ، ومخرجات اختبار (شكل ٩ - ١٠) ، ومخرجات تصحيح من DEBUGGING DECLARATIVES (شكل ٩ - ١١) .
- المخرجات غير صحيحة لما يلي :
- (١) عدد العاملين يجب أن يكون 006 وليس 012 .
- (٢) لا يوجد عامل يتساوى حاصل جمع عدد ساعات عمله المعتادة على عدد ساعات عمله وقتاً اضافياً ، مع إجمالي ساعات عمله (أي أن أعمدة شكل ٩ - ١٠ لا تجمع بطريقة صحيحة) .

```

00001 IDENTIFICATION DIVISION.
00002
00003 PROGRAM-ID. TIMELIST.
00004
00005 AUTHOR. LARRY NEWCOMER.
00006 INSTALLATION. PENN STATE UNIVERSITY--YORK CAMPUS.
00007
00008 DATE-WRITTEN. MAY 1983.
00009 DATE-COMPILED. MAY 9, 1983.
00010
00011 * TIMELIST PRINTS ONE LINE FOR EACH EMPLOYEE SHOWING:
00012 * NUMBER OF REGULAR HOURS WORKED, NUMBER OF OVERTIME
00013 * HOURS WORKED, AND TOTAL HOURS WORKED. AT THE END OF THE
00014 * REPORT, IT ALSO PRINTS A COUNT OF THE NUMBER OF
00015 * EMPLOYEES PROCESSED. INPUT IS FROM A DECK OF TIME
00016 * CARDS, KEPT IN SEQUENCE BY EMPLOYEE NAME.

00018 ENVIRONMENT DIVISION.
```

شكل (٩ - ٩)

```

00020      CONFIGURATION SECTION.

00022      SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE.
00023      OBJECT-COMPUTER. IBM-3081.

00025      INPUT-OUTPUT SECTION.

00027      FILE-CONTROL.

00029      SELECT CARD-FILE          ASSIGN TO CARDS.
00030      SELECT PRINT-FILE         ASSIGN TO PRINTER.

00032      DATA DIVISION.

00034      FILE SECTION.

00036      FD  CARD-FILE
00037      RECORD CONTAINS 80 CHARACTERS
00038      LABEL RECORDS ARE OMITTED
00039      .
00040
00041      01  TIME-CARD-INPUT          PIC X(80).

00043      FD  PRINT-FILE
00044      RECORD CONTAINS 132 CHARACTERS
00045      LABEL RECORDS ARE OMITTED
00046      .
00047
00048      01  PRINT-LINE              PIC X(132).

00050      WORKING-STORAGE SECTION.

00052      01  END-OF-CARDS-SWITCH    PIC X(3).

00054      01  WORKING-TIME-CARD.
00055
00056          05  WORKING-NAME        PIC X(20).
00057          05  WORKING-REGULAR-HOURS PIC 99.
00058          05  WORKING-OVERTIME-HOURS PIC 99.
00059          05  FILLER             PIC X(56).

00061      01  TIME-LISTING-LINE.
00062
00063          05  LISTING-NAME        PIC X(20).
00064          05  FILLER             PIC X(5)      VALUE SPACES.
00065          05  LISTING-REGULAR-HOURS PIC 99.
00066          05  FILLER             PIC X(5)      VALUE SPACES.
00067          05  LISTING-OVERTIME-HOURS PIC 99.
00068          05  FILLER             PIC X(5)      VALUE SPACES.
00069          05  LISTING-TOTAL-HOURS PIC 999.
00070          05  FILLER             PIC X(85)     VALUE SPACES.

00072      01  EMPLOYEE-TOTAL-LINE.
00073
00074          05  FILLER             PIC X(25)
00075          VALUE "NUMBER OF EMPLOYEES IS".
00076          05  NUMBER-OF-EMPLOYEES PIC 999.
00077          05  FILLER             PIC X(104)    VALUE SPACES.
00079      PROCEDURE DIVISION.

```

```

00081      DECLARATIVES.
00082
00083      CHECK-LOGIC SECTION.
00084
00085          USE FOR DEBUGGING
00086              ON ALL REFERENCES OF NUMBER-OF-EMPLOYEES
00087                  ALL REFERENCES OF WORKING-REGULAR-HOURS
00088                  ALL REFERENCES OF WORKING-OVERTIME-HOURS
00089                  ALL REFERENCES OF LISTING-TOTAL-HOURS
00090
00091          EXHIBIT NAMED DEBUG-ITEM
00092
00093
00094      END DECLARATIVES.

00096      MOVE "NO " TO END-OF-CARDS-SWITCH
00097      MOVE ZERO TO NUMBER-OF-EMPLOYEES
00098
00099      OPEN      INPUT      CARD-FILE
00100              OUTPUT      PRINT-FILE
00101
00102      PERFORM INPUT-A-RECORD
00103
00104      PERFORM PRODUCE-TIME-LISTING
00105          UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00106
00107      WRITE PRINT-LINE
00108          FROM EMPLOYEE-TOTAL-LINE
00109
00110      CLOSE CARD-FILE
00111          PRINT-FILE
00112
00113      STOP RUN

00115      INPUT-A-RECORD.
00116
00117          READ CARD-FILE
00118              INTO WORKING-TIME-CARD
00119              AT END
00120                  MOVE "YES" TO END-OF-CARDS-SWITCH
00121

00123      PRODUCE-TIME-LISTING.
00124
00125      *          INCREMENT COUNTER EACH TIME AN EMPLOYEE IS PROCESSED
00126
00127      ADD 2 TO NUMBER-OF-EMPLOYEES
00128
00129      *          MOVE DATA FROM INPUT CARD TO AREA WHERE LINE IS BUILT
00130
00131      MOVE WORKING-NAME      TO LISTING-NAME
00132      MOVE WORKING-REGULAR-HOURS TO LISTING-REGULAR-HOURS
00133      MOVE WORKING-OVERTIME-HOURS TO LISTING-OVERTIME-HOURS
00134
00135      *          CALCULATE TOTAL HOURS BY ADDING REGULAR, OVERTIME
00136      *          NOTE THAT RESULT IS PLACED IN TIME-LISTING-LINE.
00137
00138      ADD WORKING-REGULAR-HOURS WORKING-REGULAR-HOURS
00139          GIVING LISTING-TOTAL-HOURS
00140
00141      WRITE PRINT-LINE
00142          FROM TIME-LISTING-LINE
00143
00144      PERFORM INPUT-A-RECORD
00145

```

شكل (٩ - ٩) تكملة

ABEL FRED	40	03	080
BAKER SUE	30	00	060
CHARLIE CHUCK	40	12	080
PERELMAN BARNEY	40	15	080
PERELMAN MIKE	03	00	006
TROUBLE TOM	40	10	080
NUMBER OF EMPLOYEES IS	012		

شكل (٩ - ١٠)

DEBUG-ITEM = 000097	NUMBER-OF-EMPLOYEES	000
DEBUG-ITEM = 000127	NUMBER-OF-EMPLOYEES	002
DEBUG-ITEM = 000132	WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000133	WORKING-OVERTIME-HOURS	03
DEBUG-ITEM = 000138	WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000138	LISTING-TOTAL-HOURS	080
DEBUG-ITEM = 000127	NUMBER-OF-EMPLOYEES	004
DEBUG-ITEM = 000132	WORKING-REGULAR-HOURS	30
DEBUG-ITEM = 000133	WORKING-OVERTIME-HOURS	00
DEBUG-ITEM = 000138	WORKING-REGULAR-HOURS	30
DEBUG-ITEM = 000138	LISTING-TOTAL-HOURS	060
DEBUG-ITEM = 000127	NUMBER-OF-EMPLOYEES	006
DEBUG-ITEM = 000132	WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000133	WORKING-OVERTIME-HOURS	12
DEBUG-ITEM = 000138	WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000138	LISTING-TOTAL-HOURS	080
DEBUG-ITEM = 000127	NUMBER-OF-EMPLOYEES	008
DEBUG-ITEM = 000132	WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000133	WORKING-OVERTIME-HOURS	15
DEBUG-ITEM = 000138	WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000138	LISTING-TOTAL-HOURS	080
DEBUG-ITEM = 000127	NUMBER-OF-EMPLOYEES	010
DEBUG-ITEM = 000132	WORKING-REGULAR-HOURS	03
DEBUG-ITEM = 000133	WORKING-OVERTIME-HOURS	00
DEBUG-ITEM = 000138	WORKING-REGULAR-HOURS	03
DEBUG-ITEM = 000138	LISTING-TOTAL-HOURS	006
DEBUG-ITEM = 000127	NUMBER-OF-EMPLOYEES	012
DEBUG-ITEM = 000132	WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000133	WORKING-OVERTIME-HOURS	10
DEBUG-ITEM = 000138	WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000138	LISTING-TOTAL-HOURS	080

شكل (٩ - ١١)

فيما يلي إحدى طرق تصحيح البرنامج :

- ١ - تبدأ بعدد عدد العاملين . مخرجات التصحيح EXHIBIT DEBUG - ITEM على ALL REFERENCES OF NUM-BER - OF EMPLOYEES .. نجد أن NUMBER - OF EMPLOYEES قد وضع صفراً في السطر ٩٧ .
- ٢ - الإشارة التالية لـ NUMBER - OF EMPLOYEES في السطر ١٧٧ ، عندما أصبح 002 . كيف حدث قفز من 000 إلى 002 دون أخذ القيمة 001 .
- ٣ - بالنظر الى السطر ١٧٧ فإننا نعرف .

ADD 2 TO NUMBER-OF-EMPLOYEES

يفترض أن يزيد هذا السطر من العدد في كل مرة يتم إدخال سجل عامل جديد . من الواضح أننا يجب أن نجمع «1» ، وليس «2» . وما حدث هنا هو خطأ في الكتابة في برنامج الكويل . لاحظ ، على أية حال .. لم يعترض هذا الخطأ المترجم (فتكوين السطر ١٧٧ صحيح) ، ولم يتسبب في إنهاء البرنامج نهاية غير طبيعية . يجب أن يجد المبرمج مثل هذا الخطأ أثناء اختبار البرنامج .

٤ - نعود الآن إلى عمود إجمالي ساعات العمل . توضح مخرجات التصحيح محتويات WORKING - REGULAR - HOURS ، ومحتويات WORKING - OVERTIME - HOURS أثناء تنفيذ عبارة MOVE في السطرين ١٣٢ ، ١٣٣ . مثال ذلك توجد ساعات العمل المعتادة لأول عامل ، وهي 40 وساعات العمل لوقت إضافي هي 03 .

٥ - الإشارة التالية إلى WORKING - REGULAR - HOURS في السطر ١٢٨ ، والتي تشير كذلك إلى LISTING - TOTAL - HOURS ، الحقل غير الصحيح . لاحظ أن WORKING - OVERTIME - HOURS لا يشار إليه في السطر ١٢٨ ، وأن LISTING - TOTAL - HOURS تكون دائماً ضعف القيمة الموجودة في WORKING - REGULAR - HOURS .

٦ - بفحص العبارة الموجودة في السطر ١٢٨ نجد أن :

ADD WORKING-REGULAR-HOURS WORKING-REGULAR-HOURS
GIVING LISTING-TOTAL-HOURS

بدلاً من العبارة المقصودة ، وهي :

ADD WORKING-REGULAR-HOURS WORKING-OVERTIME-HOURS
GIVING LISTING-TOTAL-HOURS

وهذا خطأ آخر لا يتسبب في حدوث خطأ تكويني ، ولا ينهاي البرنامج نهاية غير طبيعية .

٩ - ١٩ صحح البرنامج التالي ، الذي يحتوى على دورة لا نهائية . قائمة برنامج المصدر ، ومخرجات الاختبار ، ومخرجات التصحيح مبينة في الأشكال : (٩ - ١٢) ، (٩ - ١٣) ، (٩ - ١٤) على التوالي .


```

00001      IDENTIFICATION DIVISION.
00002
00003      PROGRAM-ID. TIMELIST.
00004
00005      AUTHOR. LARRY NEWCOMER.
00006      INSTALLATION. PENN STATE UNIVERSITY--YORK CAMPUS.
00007
00008      DATE-WRITTEN. MAY 1983.
00009      DATE-COMPILED. MAY 9,1983.
00011      * TIMELIST PRINTS ONE LINE FOR EACH EMPLOYEE SHOWING:
00012      * NUMBER OF REGULAR HOURS WORKED, NUMBER OF OVERTIME
00013      * HOURS WORKED, AND TOTAL HOURS WORKED. AT THE END OF THE
00014      * REPORT, IT ALSO PRINTS A COUNT OF THE NUMBER OF
00015      * EMPLOYEES PROCESSED. INPUT IS FROM A DECK OF TIME
00016      * CARDS, KEPT IN SEQUENCE BY EMPLOYEE NAME.
00018      ENVIRONMENT DIVISION.
00020
00020      CONFIGURATION SECTION.
00022
00022      SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE.
00023      OBJECT-COMPUTER. IBM-3081.
00025
00025      INPUT-OUTPUT SECTION.
00027
00027      FILE-CONTROL.
00029
00029      SELECT CARD-FILE          ASSIGN TO CARDS.
00030      SELECT PRINT-FILE        ASSIGN TO PRINTER.
00032
00032      DATA DIVISION.
00034
00034      FILE SECTION.
00036
00036      FD CARD-FILE
00037      RECORD CONTAINS 80 CHARACTERS
00038      LABEL RECORDS ARE OMITTED
00039
00040
00041      01 TIME-CARD-INPUT          PIC X(80).
00043
00043      FD PRINT-FILE
00044      RECORD CONTAINS 132 CHARACTERS
00045      LABEL RECORDS ARE OMITTED
00046
00047
00048      01 PRINT-LINE              PIC X(132).
00050
00050      WORKING-STORAGE SECTION.
00052
00052      01 END-OF-CARDS-SWITCH     PIC X(3).
00054
00054      01 WORKING-TIME-CARD.
00055
00056      05 WORKING-NAME            PIC X(20).
00057      05 WORKING-REGULAR-HOURS   PIC 99.
00058      05 WORKING-OVERTIME-HOURS PIC 99.
00059      05 FILLER                 PIC X(56).

```

```

00061      01  TIME-LISTING-LINE.
00062
00063      05  LISTING-NAME          PIC X(20).
00064      05  FILLER                PIC X(5)      VALUE SPACES.
00065      05  LISTING-REGULAR-HOURS  PIC 99.
00066      05  FILLER                PIC X(5)      VALUE SPACES.
00067      05  LISTING-OVERTIME-HOURS PIC 99.
00068      05  FILLER                PIC X(5)      VALUE SPACES.
00069      05  LISTING-TOTAL-HOURS    PIC 999.
00070      05  FILLER                PIC X(85)     VALUE SPACES.

00072      01  EMPLOYEE-TOTAL-LINE.
00073
00074      05  FILLER                PIC X(25)
00075      05  NUMBER-OF-EMPLOYEES    PIC 999.      VALUE "NUMBER OF EMPLOYEES IS".
00076      05  FILLER                PIC X(104)    VALUE SPACES.
00077

00079      PROCEDURE DIVISION.

00081      DECLARATIVES.
00082
00083      CHECK-LOGIC SECTION.
00084
00085          USE FOR DEBUGGING
00086              ON ALL REFERENCES OF NUMBER-OF-EMPLOYEES
00087                  ALL REFERENCES OF WORKING-REGULAR-HOURS
00088                  ALL REFERENCES OF WORKING-OVERTIME-HOURS
00089                  ALL REFERENCES OF LISTING-TOTAL-HOURS
00090
00091      EXHIBIT NAMED DEBUG-ITEM
00092
00093
00094      END DECLARATIVES.

00096      D  READY TRACE
00097
00098      MOVE "NO " TO END-OF-CARDS-SWITCH
00099      MOVE ZERO TO NUMBER-OF-EMPLOYEES.
00100
00101      OPEN      INPUT          CARD-FILE
00102              OUTPUT          PRINT-FILE
00103
00104      PERFORM INPUT-A-RECORD
00105
00106      PERFORM PRODUCE-TIME-LISTING
00107          UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
00108
00109      WRITE PRINT-LINE
00110          FROM EMPLOYEE-TOTAL-LINE
00111
00112      CLOSE  CARD-FILE
00113          PRINT-FILE
00114
00115      STOP RUN

00117      INPUT-A-RECORD.
00118
00119      READ CARD-FILE
00120          INTO WORKING-TIME-CARD
00121          AT END
00122          MOVE "YES" TO END-OF-CARDS-SWITCH
00123

```

شكل (١٩ - ١٢) تكملة

```

00125      PRODUCE-TIME-LISTING.
00126
00127      *          INCREMENT COUNTER EACH TIME AN EMPLOYEE IS PROCESSED
00128
00129      ADD 2 TO NUMBER-OF-EMPLOYEES
00130
00131      *          MOVE DATA FROM INPUT CARD TO AREA WHERE LINE IS BUILT
00132
00133      MOVE WORKING-NAME          TO LISTING-NAME
00134      MOVE WORKING-REGULAR-HOURS TO LISTING-REGULAR-HOURS
00135      MOVE WORKING-OVERTIME-HOURS TO LISTING-OVERTIME-HOURS
00136
00137      *          CALCULATE TOTAL HOURS BY ADDING REGULAR, OVERTIME
00138      *          NOTE THAT RESULT IS PLACED IN TIME-LISTING-LINE.
00139
00140      ADD WORKING-REGULAR-HOURS WORKING-REGULAR-HOURS
00141      GIVING LISTING-TOTAL-HOURS
00142
00143      WRITE PRINT-LINE
00144      FROM TIME-LISTING-LINE
00145
00146

```

شكل (٩ - ١٢) تكملة

ABEL FRED	40	03	080
ABEL FRED	40	03	080
ABEL FRED	40	03	080
ABEL FRED	40	03	080

شكل (٩ - ١٣)

DEBUG-ITEM = 000099 NUMBER-OF-EMPLOYEES	000
INPUT-A-RECORD , PRODUCE-TIME-LISTING ,	
DEBUG-ITEM = 000129 NUMBER-OF-EMPLOYEES	002
DEBUG-ITEM = 000134 WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000135 WORKING-OVERTIME-HOURS	03
DEBUG-ITEM = 000140 WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000140 LISTING-TOTAL-HOURS	080
PRODUCE-TIME-LISTING ,	
DEBUG-ITEM = 000129 NUMBER-OF-EMPLOYEES	004
DEBUG-ITEM = 000134 WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000135 WORKING-OVERTIME-HOURS	03
DEBUG-ITEM = 000140 WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000140 LISTING-TOTAL-HOURS	080
PRODUCE-TIME-LISTING ,	
DEBUG-ITEM = 000129 NUMBER-OF-EMPLOYEES	006

شكل (٩ - ١٤)

DEBUG-ITEM = 000134 WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000135 WORKING-OVERTIME-HOURS	03
DEBUG-ITEM = 000140 WORKING-REGULAR-HOURS	40
DEBUG-ITEM = 000140 LISTING-TOTAL-HOURS	080

شكل (٩ - ١٤) تكملة

إجراء تصحيح توضيحي :

- ١ - اختبار مخرجات الاختبار : يبدو أن أول سجل مدخلات تمت طباعته مرات ومرات .
- ٢ - اختبار مخرجات التصحيح ، التي تشمل READY TRACE ، وتعرض DEBUG - ITEM على ALL REFERENCES OF NUMBER - OF - EMPLOYEES, WORKING - REGULAR - HOURS, WORKING - OVER TIME-HOURS, LISTING - TOTAL - HOURS إذا جردنا مخرجات TRACE .. فإننا نجد ما يلي :

INPUT-A-RECORD, PRODUCE-TIME-LISTING,
PRODUCE-TIME-LISTING,
PRODUCE-TIME-LISTING,
...

- من الواضح أن PRODUCE - TIME - LISTING ينفذ مرات ومرات مرة أخرى .
- ٢ - اختبار مخرجات DEBUG - ITEM لكل تنفيذ لـ PRODUCE - TIME - LISTING يظهر أن WORKING - OVERTIME - HOURS, WORKING - REGULAR - HOURS لا يتغيران ! تأكيداً لتكرار تنفيذ نفس السجل مرات ومرات .
- ٤ - بفحص البرنامج .. نجد أن السطرين ١٠٦ ، ١٠٧ يحتويان على :

PERFORM PRODUCE-TIME-LISTING
UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

- ٥ - بفحص الأسطر من ١٢٥ إلى ١٤١ .. نجد أن PRODUCE - TIME - LISTING لا يغير - END - OF - CARDS SWITCH ، ولا ينفذ PERFORM ، أى مقطع مطلوب تنفيذه ، إيجاباً .. لا توجد طريقة لتنفيذ UN- ... PERFORM TIL حتى النهاية . من الواضح كذلك أن PRODUCE - TIME - LISTING قام بتشغيل نفس السجل مرات ومرات ، حيث لا يقدم الجزء مدخلات سجل منطقي جديد .
- ٦ - يمكن تصحيح كل من المشكلتين بإضافة العبارة التالية :

PERFORM INPUT-A-RECORD

فى نهاية المقطع PRODUCE - TIME - LISTING .. تقوم INPUT - A - RECORD بوضع END - OF - CARD
PRODUCE - TIME - LIST - SWITCH فى نهاية الملف ، وكذلك وضع سجل المدخلات المنطقى التالى للتنفيذ التالى لـ
ING

الفصل العاشر

معالجة الجداول

Table Handling

سبق التعرض لمفهوم المنظومة المكون من عناصر بيانات متشابهة ، أو الجدول ، في الأقسام ١٤ من الفصل الخامس و ١٢ من الفصل السابع (انظر المثالين : ٧ - ٣٦ و ٧ - ٢٧) . وتتابع الموضوع بالتفصيل في الفصل الحالي . يقتصر اعتبارنا على جداول ذات بعد واحد وجداول ذات بعدين ، ولن نجد القارئ مشاكل في التوسع في معالجة جداول ذات بعدين ؛ لتشمل جداول ذات ثلاثة أبعاد - أكبر بعد ميسر في الكويل (انظر ملحق ح لمعرفة أكبر بعد ميسر في كويل 1985 التغطي) .

١ - ١ جداول ذات بعد واحد : جزء الحدوث

صيغة تعريف جداول ثابتة الطول ، ولها بعد واحد هي :

level-number data-name OCCURS integer TIMES

يمكن استخدام جزء الحدوث OCCURS على أي مستوى ماعدا المستوى 01 . ولكي يكون المبرمج قادرا على تحديد رقم صحيح integer .. يجب أن يعرف حجم الجدول مسبقا . وحيث إن هذا ليس مؤكدا دائما .. فمن المعتاد حجز أماكن كافية؛ لتحتوي على أقصى حجم يمكن احتياجه للجدول طوال فترة استخدام البرنامج .

مثال ١٠ - ١ :

● 01 MONTHLY-SALES-TABLE.
05 MONTHLY-SALES PIC S9(7)V99 COMP-3
OCCURS 12 TIMES.

يوجد بالتاكيد اثنا عشر شهرا في السنة .

- 01 MONTHLY-SALES-INFORMATION.
 - 05 SALESPERSON-DATA OCCURS 400 TIMES.
 - 10 SALES-ID PIC X(4).
 - 10 SALES-NAME PIC X(20).
 - 10 MONTHLY-SALES PIC S9(5)V99 COMP-3.
 - 10 QUOTA-AMOUNT PIC S9(5)V99 COMP-3.

تكرر هنا مجموعة عناصر SALESPERSON - DATA : يوجد 400 محتوى فى الجدول ، يحتوى كل منها على مجموعة من أربعة حقول جزئية على المستوى 10 . وعلى هذا ، يكون طول المحتوى 4+20+4 أى 32 بايت (انظر القسم 18 من الفصل الخامس) ، ويكون الجدول نفسه MONTHLY - SALES - INFORMATION ذا طول 400x32 : أى 12800 بايت يمثل حجم الجدول 400 تخميناً آمناً لاقصى عدد بائعين ، يجرى عليهم تشغيل أثناء فترة استخدام البرنامج .

يمكن تعريف الجداول ذات البعد الواحد (والجدول ذى الأبعاد الأخرى) كجزء من سجل فى قسم الملفات من جزء البيانات .

١ - ٢ الدلائل

بمعرفة أن جزء OCCURS يعرف عديداً من محتويات الجدول (بوصف متطابق) ... كيف يخبر المبرمج المترجم بأى المحتويات هو المطلوب فى إحدى عبارات جزء الإجراءات ؟ أبسط طريقة هى استخدام دليل subscript ، وهو رقم صحيح موجب positive integer (فى صورته ثابت عددي أو عنصر بيانات عددي) ، يكتب بين قوسين بعد اسم الجدول .

مثال ١٠ - ٢ :

تستخدم الشفرة التالية أربعة أدلة فى تجميع محتويات جدول .

```

01 QUARTERLY-SALES.
  05 QUARTER-TOTAL      PIC S9(7)V99      COMP-3
                        OCCURS 4 TIMES.
  .....
PROCEDURE DIVISION.
  .....
    ADD QUARTER-TOTAL (1)
      QUARTER-TOTAL (2)
      QUARTER-TOTAL (3)
      QUARTER-TOTAL (4)
    GIVING YEARLY-TOTAL
  
```

تظهر المنفعة الحقيقة للجداول والأدلة من الحقيقة بأنه يمكن استخدام عنصر البيانات الصحيح المتغير كدليل ، وتحدد القيمة الحالية للعنصر أى محتوى فى الجدول ، مطلوب إجراء تشغيل عليه . ومن المعتاد جدا استخدام عنصر بيانات كدليل ، يتم التحكم فيه عن طريق عبارة PERFORM ... VARYING .

مثال ١٠ - ٣ :


```

01 TOTAL-AREAS.
   05 LAST-YEARS-TOTAL PIC S9(9)V99 COMP-3.
   05 THIS-YEARS-TOTAL PIC S9(9)V99 COMP-3.
.....
01 SUBSCRIPT-AREAS.
   05 MONTH PIC S9(2) COMP SYNC.
.....
01 SALES-INFORMATION.
   05 MONTHLY-SALES OCCURS 12 TIMES.
      10 LAST-YEAR PIC S9(7)V99 COMP-3.
      10 THIS-YEAR PIC S9(7)V99 COMP-3.
.....

```

```

MOVE ZERO TO LAST-YEARS-TOTAL
THIS-YEARS-TOTAL
PERFORM CALCULATE-YEARLY-TOTALS
VARYING MONTH FROM 1 BY 1
UNTIL MONTH GREATER THAN 12
.....

```

```

CALCULATE-YEARLY-TOTALS.
  ADD LAST-YEAR (MONTH) TO LAST-YEARS-TOTAL
  ADD THIS-YEAR (MONTH) TO THIS-YEARS-TOTAL

```

هنا ، من جدول به اثنتا عشر قيمة لمجموعة العناصر MONTHLY - SALES ، وهى اثنتا عشر قيمة للعنصر الفردى LAST-YEAR ، واثنتا عشر قيمة للعنصر الفردى THIS - YEAR ، تجمع باستخدام دليل متغير MONTH .
 فى كويل IBM OS /VS .. الطريقة الأكثر كفاءة فى تعريف عنصر بيانات عددي لاستخدامه كدليل ، هى PIC S9 (n) .
 COMP SYNC .. الدلائل مطلوبة لكل عناصر البيانات عند ، أو تابعة لـ ، جزء OCCURS .

١ - ٣ معالجة جداول ذات بعد واحد

تحميل (وضع قيم ابتدائية) ، جدول بأصفار

يتكرر طلب وضع قيم ابتدائية لكل محتويات الجدول بنفس القيمة ، ولتكن صفر . ونظرا لعدم إمكانية استخدام جزء VALUE مع عناصر البيانات التى تكون جزءا من جدول .. فإن عبارة MOVE وسيلة بسيطة لوضع الأصفار فى الجدول.

مثال ١٠ - ٤ :

لتعريف جدول به 50 عنصر كما يلى :

```

01 SALES-FIGURES-BY-STATE.
   05 STATE-SALES OCCURS 50 TIMES.
   10 STATE-ID PIC X(2).
   10 SALES-AMOUNT PIC S9(5)V99 COMP-3.

```

فإننا نحدد طريقة غير صحيحة ، والطريقة الصحيحة ، لوضع أصفار فى الجدول .

- MOVE ZEROS TO SALES-FIGURES-BY-STATE

عبارة صحيحة .. طالما أن SALES - FIGURES - BY - STATE ليست عنصر جدول (فهي ليست موصوفة كجزء OCCURS ، وليست تابعة لجزء OCCURS) ، وبالتالي .. فلا تتطلب دليلاً ، وحيث إنها مجموعة عناصر لها الشكل PIC X DISPLAY .. فإن أسفاراً إلى محتويات الجدول ، وبصفة خاصة ، في 50 حدث لـ SALES - AMOUNT ، وهي عنصر من نوع 3 - COMP . وعلى هذا .. يوجد خطأ منطقي .

```

●      PERFORM ZERO-SALES-AMOUNT
        VARYING STATE-SALES-SUB
        FROM 1 BY 1
        UNTIL STATE-SALES-SUB GREATER THAN 50
        .....
ZERO-SALES-AMOUNT.
        MOVE ZERO TO SALES-AMOUNT (STATE-SALES-SUB)
        MOVE SPACES TO STATE-ID (STATE-SALES-SUB)
    
```

الإجراء الصحيح هو تعريف دليل (STATE - SALES - SUB) ، حيث يمكن توضيح الحقول الجزئية للجدول في مقطع PERFORM .

مثال ١٠ - ٥ :

```

01 EMPLOYEE-DATA-TABLE.
    05 EMPLOYEE-INFORMATION OCCURS 500 TIMES.
        10 SOCIAL-SEC-NUMBER PIC 9(9).
        10 EMPLOYEE-SALARY PIC S9(3)V99.
    
```

نظراً لأن كل الحقول من نوع DISPLAY .. فيمكننا أن نضع أسفاراً في كل محتويات الجدول بعبارة واحدة ، وهي :

```
MOVE ZEROS TO EMPLOYEE-DATA-TABLE
```

التحميل بجزء VALUE

إذا كان من الواجب وضع قيمة ابتدائية خاصة بكل عنصر من عناصر الجدول .. فإنه يمكن استخدام جزء VALUE في إعداد مجموعة من عناصر البيانات التي يمكن بعد ذلك إعادة تعريفها REDEFINES كجدول . (الدوران ضروري لأن جزء VALUE لا يمكن تطبيقه مباشرة على عناصر الجدول) . يمكن استخدام جزء VALUE بهذه الطريقة مع عناصر مخزن العمل.

مثال ١٠ - ٦ :

```

01 VALID-STATUS-CODES-TABLE.
    05 VALID-STATUS-VALUES.
        10 FILLER PIC X(3) VALUE "A07".
        10 FILLER PIC X(3) VALUE "A2K".
        10 FILLER PIC X(3) VALUE "B03".
        10 FILLER PIC X(3) VALUE "C99".
    05 STATUS-CODE REDEFINES VALID-STATUS-VALUES
        PIC X(3)
        OCCURS 4 TIMES.
    
```

يحتوى VALID - STATUS - VALUES على أربعة عناصر فردية ، لكل منها (3) PIC X واستخدام جزء VALUE فى وضع قيم ابتدائية لها .

لاحظ أننا لا نهتم بإعطاء أسماء لعناصر البيانات هذه ، ولكن نستخدم الكلمة المحجوزة FILLER بدلا من ذلك . وعند ذلك .. يعاد تعريف VALID - STATUS - VALUES كجدول ، STATUS - CODE ، بطريقة تجعل العناصر المعرفة بانها (3) PIC X فى STATUS - CODE والعناصر الأربعة المعرفة بانها (3) PIC X فى VALID - STATUS - VALUES تكون هي نفسها (أى إنها تحتل نفس مواقع الذاكرة تماما) . وهذا يعنى أن (1) STATUS - CODE له قيمة بداية A07 و (2) STATUS - CODE له قيمة بداية A2K ... وهكذا . وتشير العبارات التى تجرى تشغيلا بالفعل على الجدول ، بالطبع ، إلى STATUS - CODE ، وليس إلى VALID - STATUS - VALUES .

مثال ١٠ - ٧ :

استخدم مثال ١٠ - ٦ جزء VALUE فى وضع قيم ابتدائية فى جدول به عناصر فردية ، والآن نضع قيما ابتدائية فى جدول به مجموعات عناصر .

```
01 SHIPPING-INSTRUCTIONS-TABLE.
05 SHIPPING-CODES-VALUES.
    10 FILLER                PIC X(2)    VALUE "A3".
    10 FILLER                PIC X(20)   VALUE "U.S. MAIL".
    10 FILLER                PIC X(2)    VALUE "A8".
    10 FILLER                PIC X(20)   VALUE "UPS".
    10 FILLER                PIC X(2)    VALUE "C3".
    10 FILLER                PIC X(20)   VALUE "FEDERAL EXPRESS".
05 CODES-AND-INSTRUCTIONS REDEFINES SHIPPING-CODES-VALUES
    OCCURS 3 TIMES.
    10 SHIPPING-CODE        PIC X(2).
    10 SHIPPING-INSTRUCTION PIC X(20).
```

مرة أخرى .. تعرف عناصر الجدول كعناصر فردية أولا باستخدام اسم FILLER ، وأجزاء PIC ، USAGE ، VALUE المناسبة . لاحظ أن SHIPPING - CODES - VALUES تحتوى على عنصر له (2) PIC X ، يتبعه عنصر له (20) PIC X ، يتبعه عنصر ثانٍ له (1) PIC X وهكذا . ويقوم CODES - AND - INSTRUCTIONS بإعادة تعريف SHIPPING - CODES - VALUES كجدول به ٣ عناصر ؛ حيث يحتوى كل عنصر SHIPPING - CODE له (2) PIC X ، يليه SHIPPING - INSTRUCTION له (20) PIC X . بسبب أجزاء VALUE .. فإن (1) SHIPPING - CODE له قيمة بداية "A3" ، وتكون له (1) SHIPPING - INSTRUCTION قيمة بداية "U.S. MAIL" ، وتكون له (2) SHIPPING - INSTRUCTION قيمة بداية "A8" كما أن (2) SHIPPING - INSTRUCTION تكون له قيمة بداية "USP" .. وهكذا . يطبق مثال (١٠ - ٢٠) هذا الأسلوب على جدول به عناصر عديدة .

التحميل بعبارة READ

هناك عيبان من وضع القيم الابتدائية فى جداول مخزن العمل عن طريق إعادة تعريف العناصر بعناصر لها جزء VALUE : (١) مع ازدياد حجم الجداول وتعقيدها .. تصبح هذه الطريقة مزعجة جدا وتتطلب شفرة كثيرة فى جزء البيانات (٢)

إذا كان لابد من تغيير القيم الابتدائية لعناصر الجدول .. فيجب تعديل برنامج الكويل نفسه وإعادة ترجمته . وهناك طريقة أخرى فعالة لتجنب هذه الصعوبات ، وهي حفظ قيم بداية الجدول في ملف قرص تتابعي أو ملف شريط تتابعي . وعندما يبدأ تنفيذ البرنامج .. يمكنه أن يقرأ READ القيم الابتدائية في المناطق المناسبة من الجدول . فإذا كانت هناك حاجة لتغيير قيم البداية للجدول .. فيمكن عمل ذلك بسهولة عن طريق تعديل الملف .

مثال ١٠ - ٨ :

جدول رموز الحالة الخاص بـمثال (١٠ - ٦) من ملف بالطريقة التالية :

```
FD STATUS-CODE-FILE ...
01 STATUS-CODE-RECORD.
   05 STATUS-CODE-VALUE          PIC X(3).
   .....

WORKING-STORAGE SECTION.
   .....

01 VALID-STATUS-CODES-TABLE.
   05 STATUS-CODE                PIC X(3)  OCCURS 400 TIMES.
   05 NUMBER-OF-CODES            PIC S9(3)  COMP SYNC.
   .....

PROCEDURE DIVISION.
  OPEN INPUT STATUS-CODE-FILE
  MOVE "NO" TO END-STATUS-FILE-SW
  PERFORM LOAD-STATUS-TABLE
    VARYING NUMBER-OF-CODES FROM 1 BY 1
    UNTIL      END-STATUS-FILE-SW EQUAL "YES"
           OR  NUMBER-OF-CODES GREATER THAN 400
  IF NUMBER-OF-CODES GREATER THAN 400
    MOVE 400 TO NUMBER-OF-CODES
    DISPLAY "TABLE FILE TOO LARGE--MODIFY, RECOMPILE COBOL TABLE"
    UPON OPERATOR-MESSAGE-DEVICE
  ELSE
    SUBTRACT 1 FROM NUMBER-OF-CODES

  CLOSE STATUS-CODE-FILE
  .....

LOAD-STATUS-TABLE.
  READ STATUS-CODE-FILE
  AT END
    MOVE "YES" TO END-STATUS-FILE-SW

  IF END-STATUS-FILE-SW EQUAL "NO"
    MOVE STATUS-CODE-VALUE TO STATUS-CODE (NUMBER-OF-CODES)
```

لاحظ كيف استخدم NUMBER - OF - CODES في عدد عناصر الجدول التي أدخلت فعلا من الملف ، ولاحظ أن دورة PERFORM ... VARYING التي تسجل قيم شفرة الحالة تتوقف عند نهاية الملف ، أو إذا تعدى - NUMBER - OF - CODES حجم الجدول ، كما هو موضح في جزء OCCURS (400 في هذه الحالة) . عبارة IF تلى .. PERFORM VARYING تختبر عند ذلك NUMBER - OF - CODES ، وإذا تعدى أقصى حجم للجدول ، تعرض رسالة لمشغل الكمبيوتر ، التي تحدد أن جزء OCCURS يجب أن يتغير وتعاد ترجمة البرنامج . لاحظ إنه إذا حدثت نهاية الملف .. فإن آخر قيمة (NUMBER - OF - CODES تناظر نهاية الملف ، وليست ل STATUS - CODE - VALUE . وعلى هذا .. يجب أن يقل NUMBER - OF - CODES بمقدار 1 ، بعد توقف PERFORM ... VARYING عند نهاية الملف . وفي أى حالة .. يحدد NUMBER - OF - CODES - بطريقة صحيحة - عدد عناصر الجدول المحملة لأى مرة من مرات تنفيذ برنامج معين . إن حفظ عداد بعدد العناصر النشطة في الجدول شائع الاستخدام جدا كوسيلة في معالجة الجداول .

تجميع عناصر الجدول

بالإضافة إلى مثال ١٠ - ٣ فإننا نضرب مثالا آخر .

مثال ١٠ - ٩ :

أفرض أنه مطلوب تجميع مجموع رواتب العاملين في جدول مثال (١٠ - ٥) .

```
01 EMPLOYEE-DATA-TABLE.
05 EMPLOYEE-INFORMATION OCCURS 500 TIMES.
10 SOCIAL-SEC-NUMBER PIC 9(9).
10 EMPLOYEE-SALARY PIC S9(3)V99 COMP-3.
05 NUMBER-ACTIVE-EMPLOYEES PIC S9(3) COMP.
.....
MOVE ZERO TO TOTAL-SALARIES
PERFORM TOTAL-EMPLOYEE-SALARIES
VARYING EMPLOYEE-SUB
FROM 1 BY 1
UNTIL EMPLOYEE-SUB GREATER THAN NUMBER-ACTIVE-EMPLOYEES
.....
TOTAL-EMPLOYEE-SALARIES.
ADD EMPLOYEE-SALARY (EMPLOYEE-SUB) TO TOTAL-SALARIES
```

لم نجمع هنا كل 500 عنصر ، فبالرغم من أن الجدول معرف ليحتوى 500 عامل كحد أقصى ، إلا أنه قد لا يحتوى إلا على بيانات لعدد أقل من 500 عامل . يستخدم عنصر البيانات NUMBER - ACTIVE - EMPLOYEES (وهو بنفسه لا يعد جزءا من الجدول) في حفظ تتبع عدد العاملين الممثلين فعلا في الجدول . ويفترض أن الأجزاء التي تضيف عاملين جدد الى الجدول ، أو تحذف عاملين منه تظل محتفظة بهذا العدد كذلك . يصبح NUMBER - ACTIVE - EMPLOYEES قيمة إيقاف عبارة PERFORM التي تتحكم في جمع رواتب العاملين .

فحص الجدول باستخدام PERFORM

في عملية الفحص look - up أو البحث search .. تقارن عناصر الجدول مع قيمة معينة حتى يوجد العنصر المتوافق مع هذه القيمة ، أو حتى يتقرر عدم وجود مثل هذه القيمة في الجدول . وأبسط خوارزمي للبحث في جدول ، هو البحث التتابعى se-quential أو الخطى linear : حيث يفحص فيه أول عنصر ... فالثاني ، فالثالث وهكذا ، حتى يتم الوصول الى العنصر المطلوب ، أو ينتهى الجدول . توفر معظم صيغ الكويل عبارة بحث SEARCH (انظر القسم الثامن من هذا الفصل) لبرمجة البحث التتابعى ، ولكنه يمكن كذلك عمل البرمجة باستخدام دورة PERFORM .

مثال ١٠ - ١٠ :

بمعرفة الجدول المعطى في مثال (١٠ - ٦) ، وقيمة معروفة في (3) PIC X - STATUS - CODE INPUT : فالمطلوب تحديد ما اذا كان INPUT - STATUS - CODE صحيحا ، وذلك بفحص الجدول VALID - STATUS - CODE - TABLE .

```
01 VALID-STATUS-CODES-TABLE.
   05 STATUS-CODE          PIC X(3)
                               OCCURS 40 TIMES.
   .....
CHECK-STATUS-CODE.
  PERFORM LOOK-UP-STATUS
  IF VALID-STATUS-SW EQUAL "YES"
  ... (process valid status code)
  ELSE
  ... (handle invalid status code)
  .....
LOOK-UP-STATUS.
  MOVE "NO" TO VALID-STATUS-SW
  PERFORM CHECK-ENTRY
    VARYING STATUS-CODE-SUBSCRIPT FROM 1 BY 1
    UNTIL STATUS-CODE-SUBSCRIPT GREATER THAN 40
    OR VALID-STATUS-SW EQUAL "YES"

CHECK-ENTRY.
  IF STATUS-CODE (STATUS-CODE-SUBSCRIPT) EQUAL INPUT-STATUS-CODE
    MOVE "YES" TO VALID-STATUS-SW
```

يضع LOOK - UP - STATUS القيمة "NO" في VALID - STATUS - SW أولا . وينفذ CHECK - ENTRY بعد ذلك ، مع تغيير STATUS - CODE - SUBSCRIPT (المعروف على أنه PIC S9(5) COMP SYNC) من 1 حتى عدد عناصر الجدول ؛ فإذا تساوى أحد عناصر الجدول المناظر للدليل STATUS - CODE - SUBSCRIPT - STATUS - CODE - في أى لحظة - مع INPUT - STATUS - CODE ، فتوضع YES في VALID - STATUS - SW ،

وعلى هذا .. يتوقف التنفيذ لعبارة PERFORM ... UNTIL . فإذا لم يتساوى أى عنصر من عناصر الجدول مع INPUT STATUS - CODE - ، فإن القيمة "NO" تظل موجودة في VALID - STATUS - SW .

لاحظ إنه اذا تساوى العنصر السادس (مثلا) مع INPUT - STATUS - CODE .. تتوقف عبارة PERFORM ، مع مساواة STATUS - CODE - SUBSCRIPT بالرقم ٧ (شكل ٧ - ١٤) . ويمكن معالجة هذا القصور كما يلي :

```

LOOK-UP-STATUS.
  MOVE 1 TO STATUS-CODE-SUBSCRIPT
  MOVE "YES" TO VALID-STATUS-SW
  PERFORM CHECK-ENTRY
    UNTIL VALID-STATUS-SW EQUAL "NO"
      OR STATUS-CODE (STATUS-CODE-SUBSCRIPT) EQUAL
        INPUT-STATUS-CODE

```

```

CHECK-ENTRY.
IF STATUS-CODE-SUBSCRIPT EQUAL 40
  MOVE "NO" TO VALID-STATUS-SW
ELSE
  ADD 1 TO STATUS-CODE-SUBSCRIPT

```

والآن إذا أشار STATUS - CODE - SUBSCRIPT إلى عنصر بأنه يساوى INPUT - STATUS - CEOD .. فإن شرط UNTIL يتحقق وينتهي تنفيذ PERFORM فوراً دون زيادة STATUS - CODE - SUBSCRIPT والذي يشير إلى العنصر المطلوب في الجدول .

مثال ١٠ - ١١ :

هناك سبب منتشر لاستخدام البحث في الجدول ، وهو فك شفرة decode حقول مدخلات ؛ لطباعتها في تقارير ، ونوضح ذلك بجدول يشبه جدول مثال (٧ - ١٠) .

```

01 SHIPPING-INSTRUCTIONS-TABLE.
05 CODES-AND-INSTRUCTIONS OCCURS 18 TIMES.
10 SHIPPING-CODE PIC X(2).
10 SHIPPING-INSTRUCTION PIC X(30).
.....
  MOVE "YES" TO FOUND-CODE-SW
  MOVE 1 TO SHIPPING-SUBSCRIPT
  PERFORM CHECK-ENTRY
    UNTIL SHIPPING-CODE (SHIPPING-SUBSCRIPT) EQUAL INPUT-CODE
      OR FOUND-CODE-SW EQUAL "NO"
  IF FOUND-CODE-SW EQUAL "YES"
    MOVE SHIPPING-INSTRUCTION (SHIPPING-SUBSCRIPT) TO
      PRINTED-SHIPPING-MESSAGE
  ELSE
    MOVE "*** INVALID SHIPPING CODE ***" TO
      PRINTED-SHIPPING-MESSAGE
.....
CHECK-ENTRY.
  IF SHIPPING-SUBSCRIPT EQUAL 18
    MOVE "NO" TO FOUND-CODE-SW
  ELSE
    ADD 1 TO SHIPPING-SUBSCRIPT

```

إذا وجد INPUT - CODE في SHIPPING - INSTRUCTION - TABLE فإن SHIPPING - INSTRUCTION - TABLE - CODE الذي يتفق مع INPUT - CODE ينقل إلى منطقة طباعة المخرجات ؛ فإذا لم يوجد INPUT - CODE في الجدول .. فتطبع رسالة خطأ في التقرير . وتسمح هذه الطريقة بأن تحتوى سجلات المدخلات على رمز من 2 بايت فقط ، مقدمة على ذلك تعليمات الشحن المكون من 30 بايت في تقرير الطباعة .

١٠ - ٤ جداول ذات بعدين

عندما تعرف عناصر تابعة لأحد عناصر OCCURS لها جزء OCCURS أيضا .. فإننا نقول إن لهذا الجدول بعدين two dimensions .

مثال ١٠ - ١٢ :

01 FREIGHT-CHARGES-TABLE.		
05 WEIGHT-RANGE	OCCURS 3 TIMES.	
10 WEIGHT-LIMIT	PIC S9(3)	COMP-3.
10 DESTINATION	OCCURS 3 TIMES.	
15 FREIGHT-CHARGE	PIC S9(3)V99	COMP-3.

WEIGHT - RANGE هو جدول به ٣ عناصر ، ويحتوى كل عنصر على الحقول DESTINATION , WEIGHT - RANGE LIMIT ... إلا أن DESTINATION نفسه عبارة عن جدول به ٣ عناصر (يحتوى كل منها على قيمة - FREIGHT CHARGE) . وعلى هذا .. فالجدول التابع DESTINATION يحتوى على $3 \times 3 = 9$ عناصر ، انظر شكل (١٠ - ١) ، لاحظ أن عدد الدلائل المطلوب للإشارة إلى أحد عناصر الجدول يساوى عدد أجزاء OCCURS المستخدمة مع العنصر . يشير أول دليل إلى أعلى مستوى لجزء OCCURS ، ويشير الدليل الثانى للمستوى التالى له ... وهكذا . وبالنسبة لهذا الجدول الخاص الذى له بعدان .. فإن الدليلين لهما نفس مدى القيم ، وأن يكون هذا صحيحا بالطبع لكل الجداول . فى شكل (١٠ - ١) .. تفصل الدلائل المتعددة بواسطة فاصلة وفراغ ، الفاصلة اختيارية .

WEIGHT-RANGE (1)	WEIGHT-LIMIT (1)	2 bytes, S9(3)	COMP-3
	FREIGHT-CHARGE (1, 1)	3 bytes, S9(3)V99	COMP-3
	FREIGHT-CHARGE (1, 2)	3 bytes, S9(3)V99	COMP-3
	FREIGHT-CHARGE (1, 3)	3 bytes, S9(3)V99	COMP-3
WEIGHT-RANGE (2)	WEIGHT-LIMIT (2)	2 bytes, S9(3)	COMP-3
	FREIGHT-CHARGE (2, 1)	3 bytes, S9(3)V99	COMP-3
	FREIGHT-CHARGE (2, 2)	3 bytes, S9(3)V99	COMP-3
	FREIGHT-CHARGE (2, 3)	3 bytes, S9(3)V99	COMP-3
WEIGHT-RANGE (3)	WEIGHT-LIMIT (3)	2 bytes, S9(3)	COMP-3
	FREIGHT-CHARGE (3, 1)	3 bytes, S9(3)V99	COMP-3
	FREIGHT-CHARGE (3, 2)	3 bytes, S9(3)V99	COMP-3
	FREIGHT-CHARGE (3, 3)	3 bytes, S9(3)V99	COMP-3

شكل (١٠ - ١)

مثال ١٠ - ١٣ :

إذا كان السجل STUDENT - TRANSCRIPT - المعرف أدناه سجلات - منطقيا ملف الطلبة الرئيسي ، فما طول السجل المنطقي ؟

```

01 STUDENT-TRANSCRIPT.
   05 STUDENT-ID          PIC X(9).
   05 STUDENT-NAME        PIC X(25).
   05 SEMESTERS-ATTENDED  PIC S9          COMP-3.
   05 SEMESTER-INFORMATION OCCURS 8 TIMES.
       10 GRADE-POINT-AVERAGE PIC S9V99    COMP-3.
       10 NUMBER-COURSES      PIC S9          COMP-3.
       10 COURSE-INFO         OCCURS 6 TIMES.
           15 COURSE-ID       PIC X(5).
           15 NUMBER-CREDITS  PIC S9V9       COMP-3.
           15 GRADE           PIC S9          COMP-3.

```

تعريف المقرر ID - COURSE طوله 5 بايت ، وعدد الساعات NUMBER - CREDITS (القسم ١٨ - من الفصل الخامس) طوله 2 بايت وطول الدرجة GRADE هو بايت واحد ، وعلى هذا ، فمختصر معلومات المقرر COURSE - INFO طوله ٨ بايت ، وحيث إن COURSE - INFO OCCURS .. فإن محتويات الجدول COURSE - INFO يبلغ طولها 48 بايت. يحتوى عنصر SEMESTER - INFORMATION على محتويات الجدول COURSE - INFO ؛ مضافا إليها - NUMBER COURSES (طوله بايت واحد) وكذلك GRADE - POINT - AVERAGE (طوله 2 بايت) ، وعلى هذا .. يصبح طول SE-

SEMESTER - INFORMATION عدد 51 بايت،. وحيث إنه هناك 8 عناصر تتطلب محتويات الجدول - IN - SEMESTER FORMATION عدد 408 بايت . ويجب أن نجمع عليها مكانا للحقل لرقم تعريف الطالب ID - STUDENT (9 بايت)، ومكانا آخر لاسم الطالب NAME - STUDENT (25 بايت) ، ومكانا آخر لعدد الفصول الدراسية التي درسها - SEMESTERS ATTENDED (واحد بايت) ، ويصبح إجمالي طول السجل 443 بايت .

يجب ملاحظة أن تعريف STUDENT - TRANSCRIPT يفترض مسبقا تحميل الجدول ؛ بحيث يشير - COURSE ، INFO (i,j) إلى المقرر رقم j الذي درسه الطالب أثناء الفصل i خلال دراسته (أي أنه إذا كان الجدول معروضا على هيئة مصفوفة .. فنجد أنه يضغط في اتجاه الركن العلوي الأيسر) . يخزن عدد الفصول الدراسية التي درسها الطالب في العداد SEMESTER - ATTENDED ، كما يخزن العداد NUMBER - COURSES (k) عدد المقررات التي درسها الطالب أثناء الفصل الدراسي k في دراسته . تحدد هذه العدادات أيًا من مواقع ذاكرة COURSE - INFO البالغ عددها ٤٨ (٨ × ٦) بالضبط ، يحتوى على بيانات نشطة .

مثال ١٠ - ١٤ :

أحسب متوسط اداء الفصل الدراسي لطالب معلوماته موضوعه حاليا في السجل المنطقي - STUDENT TRANSCRIPT الموجود في مثال (١٠ - ١٣) .

```
MOVE ZERO TO GRADE-POINT-TOTAL
PERFORM ADD-UP-SEMESTER-AVERAGES
  VARYING SEMESTER-SUB FROM 1 BY 1
  UNTIL SEMESTER-SUB GREATER THAN SEMESTERS-ATTENDED
DIVIDE SEMESTERS-ATTENDED INTO GRADE-POINT-TOTAL
  GIVING AVERAGE-SEMESTER-GRADE
.....
ADD-UP-SEMESTER-AVERAGES.
  ADD GRADE-POINT-AVERAGE (SEMESTER-SUB) TO GRADE-POINT-TOTAL
```

١ - ٥ معالجة جداول ذات بعدين

تحميل أصفار

حيث إنه يستخدم دليان ... فتستخدم عبارة ... AFTER ... VARYING ... PERFORM

مثال ١٠ - ١٥ :

يفترض فيما يلي أن كل محتويات الجدول الستة نشطة حالياً .

```

01 SALES-AMOUNTS.
  05 REGION-TOTALS      OCCURS 2 TIMES.
    10 DEPARTMENT-TOTALS OCCURS 3 TIMES.
      15 SALES           PIC S9(5)V99  COMP.

```

```

01 SUBSCRIPTS.
  05 REGION-NUMBER      PIC S9(5)  COMP SYNC.
  05 DEPARTMENT-NUMBER  PIC S9(5)  COMP SYNC.

```

```

PERFORM ZERO-SALES-AMOUNTS
  VARYING REGION-NUMBER FROM 1 BY 1
    UNTIL REGION-NUMBER GREATER THAN 2
  AFTER DEPARTMENT-NUMBER FROM 1 BY 1
    UNTIL DEPARTMENT-NUMBER GREATER THAN 3

```

```

ZERO-SALES-AMOUNTS.
  MOVE ZERO TO SALES (REGION-NUMBER DEPARTMENT-NUMBER)

```

جمع المحتويات

مثال ١٠ - ١٦ :

احسب إجمالي الساعات حتى الآن للطالب في مثال (١٠ - ١٣) .

```

05 TOTAL-CREDITS      PIC S9(3)V9  COMP-3.

```

```

MOVE ZERO TO TOTAL-CREDITS
PERFORM ACCUMULATE-TOTAL-CREDITS
  VARYING SEMESTER-NUMBER FROM 1 BY 1
    UNTIL SEMESTER-NUMBER GREATER THAN SEMESTERS-ATTENDED
  AFTER COURSE-NUMBER FROM 1 BY 1
    UNTIL COURSE-NUMBER GREATER THAN
      NUMBER-COURSES (SEMESTER-NUMBER)

```

```

ACCUMULATE-TOTAL-CREDITS.
  ADD NUMBER-CREDITS (SEMESTER-NUMBER COURSE-NUMBER)
    TO TOTAL-CREDITS

```

حيث تعرف الأدلة كما يلي :

```
01 SUBSCRIPTS.
05 SEMESTER-NUMBER PIC S9(5) COMP SYNC.
05 COURSE-NUMBER PIC S9(5) COMP SYNC.
```

لاحظ بصفة خاصة الأدلة الخاصة بالعنصر NUMBER - COURSES في عبارة PERFORM. ونظرا لتكرار حدوث هذا العنصر ٨ مرات - كجزء من الجدول SEMESTER - INFORMATION - يجب أن تكون له أدلة. وفي كل مرة يزداد SEMESTER - NUMBER .. تتغير قيمة (SEMESTER - NUMBER) COURSES طبقا لهذه الزيادة.

مثال ١٠ - ١٧ :

في مثال (١٠ - ١٣)، ومثال (١٠ - ١٤) نجد أن محتويات GRADE - POINT - AVERAGE ، وهي جزء من جدول SEMESTER - INFORMATION سبق معاملتها كيانات مدخلات. افترض أنه من الضروري حساب GRADE - POINT - AVERAGE. من جدول COURSE - INFO : يمكن أن يحدث هذا ، لكل فصل دراسي ، بتجميع الدرجات مضروبة في عدد الساعات لكل المقررات التي درست خلال الفصل الدراسي وقسمتها على إجمالي عدد الساعات التي درست خلال هذا الفصل الدراسي ، ويمكن أن تشمل شفرة الكويل هيكلًا VARYING ... PERFORM متداخلا ، كما هو مبين أدناه ، حيث الأدلة كما هي معرفة في مثال (١٠ - ١٦) ، وحيث العناصر TOTAL - GRADE - POINTS ، TOTAL - CREDITS ، هي عناصر عديدة في مخزن العمل .

```
PERFORM CALCULATE-SEMESTER-AVERAGE
  VARYING SEMESTER-NUMBER FROM 1 BY 1
  UNTIL SEMESTER-NUMBER GREATER THAN SEMESTERS-ATTENDED
.....
CALCULATE-SEMESTER-AVERAGE.
  MOVE ZERO TO TOTAL-GRADE-POINTS
  TOTAL-CREDITS
  PERFORM TOTAL-SEMESTER-COURSES
    VARYING COURSE-NUMBER FROM 1 BY 1
    UNTIL COURSE-NUMBER GREATER THAN
      NUMBER-COURSES (SEMESTER-NUMBER)
  DIVIDE TOTAL-CREDITS INTO TOTAL-GRADE-POINTS
    GIVING GRADE-POINT-AVERAGE (SEMESTER-NUMBER)

TOTAL-SEMESTER-COURSES.
  ADD NUMBER-CREDITS (SEMESTER-NUMBER COURSE-NUMBER)
    TO TOTAL-CREDITS
  COMPUTE TOTAL-GRADE-POINTS = TOTAL-GRADE-POINTS +
    GRADE (SEMESTER-NUMBER COURSE-NUMBER) *
    NUMBER-CREDITS (SEMESTER-NUMBER COURSE-NUMBER)
```

تحويل بيانات مدخلات الى دلائل

بالنسبة للجداول التي تعرضنا لها حتى الآن ، كانت الدلائل تنتج بواسطة ... VARYING ... PERFORM ، ولكن الحصول على قيم دلائل عن طريق تحويل بيانات مدخلات يكون مطلوباً في بعض الأحيان . أبسط تحويل هو عدم التحويل - وذلك عندما تكون الحالة بيانات المدخلات (طبقاً لتصميمها) مناسبة كما هي .
مثال ١٠ - ١٨ :

افترض أن العامل في إدخال البيانات .. أدخل حقول DISPLAY التالية :

```
FD SALES-INPUT-FILE...
01 SALES-RECORD.
   05 REGION-ID      PIC 9(1).
   05 DEPARTMENT-ID  PIC 9(2).
   05 SALES-AMOUNT   PIC S9(4)V99.
```

ينتج عن استخدام REGION - ID و DEPARTMENT - ID كدلائل جدول لإجمالي المبيعات ؛ طبقاً للمنطقة والقسم.

```
01 TOTAL-SALES-AMOUNTS-TABLE.
   05 REGION              OCCURS 7 TIMES.
   10 DEPARTMENT          OCCURS 12 TIMES.
   15 TOTAL-SALES-AMOUNT  PIC S9(5)V99 COMP.
   05 REGION-LIMIT        PIC S9(2)    COMP SYNC VALUE + 7.
   05 DEPARTMENT-LIMIT    PIC S9(2)    COMP SYNC VALUE + 12.
```

يمكن أن تعمل الشفرة التالية (يفترض أن TOTAL - SALES - AMOUNT قد سبق وضع أصفار في محتوياتها) :

```
.....
OPEN INPUT SALES-INPUT-FILE
MOVE "NO" TO END-FILE-SW
PERFORM GET-SALES-RECORD
PERFORM ACCUMULATE-SALES-TOTALS
  UNTIL END-FILE-SW EQUAL "YES"
CLOSE SALES-INPUT-FILE
.....
```

```
GET-SALES-RECORD.
  READ SALES-INPUT-FILE
  AT END
    MOVE "YES" TO END-FILE-SW
```

```
ACCUMULATE-SALES-TOTALS.
  IF REGION-ID NOT NUMERIC OR DEPARTMENT-ID NOT NUMERIC OR
    SALES-AMOUNT NOT NUMERIC
    PERFORM INVALID-NUMBER-ROUTINE
  ELSE IF REGION-ID LESS THAN 1 OR GREATER THAN REGION-LIMIT
    PERFORM REGION-OUT-OF-RANGE-ROUTINE
  ELSE IF DEPARTMENT-ID LESS THAN 1 OR GREATER THAN
    DEPARTMENT-LIMIT
    PERFORM DEPARTMENT-OUT-OF-RANGE-ROUTINE
  ELSE
    ADD SALES-AMOUNT TO
      TOTAL-SALES-AMOUNT (REGION-ID DEPARTMENT-ID)
  PERFORM GET-SALES-RECORD
```

لاحظ التأكد من عناصر المداخل المستخدمة كدلائل ، لاتقدم معظم صيغ الكويل أى تحذير تلقائى ، وذلك عندما تقع إشارة بدلائل خارج الجدول .

يمكن أن يتحقق التحويل الفعلى لبيانات المداخل باستخدام عبارة IF خطية ، على أن يكون عدد القيم صغيرا نسبيا ، وإلا تحقق ذلك بواسطة فحص الجدول . يوضح مثال (١٠ - ١٩) كلاً من الطريقتين .

مثال ١٠ - ١٩ :

أعد حل مثال (١٠ - ١٨) ، إذا كان حجم الجدول 3x15 (بدلاً من 7x12) . وإذا كانت الحقول الحرفية عديدة معطاة على النحو التالى :

```
01 SALES-RECORD.
05 REGION-ID          PIC X(3).
05 DEPARTMENT-ID      PIC X(5).
05 SALES-AMOUNT       PIC S9(4)V99.
```

يمكننا استخدام جدول ذى بعد واحد فى تحويل DEPARTMENT - ID الى الدليل DEPARTMENT - NUMBER .

```
01 DEPARTMENT-ID-TABLE.
05 VALID-DEPARTMENT-ID OCCURS 15 TIMES
PIC X(5).
```

يتحول أول عنصر فى جدول DEPARTMENT - ID - TABLE الى القيمة 1 من DEPARTMENT - NUMBER . والعنصر الثانى الى القيمة 2 وهكذا . وعلى هذا ... يمكننا أن نحصل على DEPARTMENT - NUMBER بالبحث عن DEPARTMENT - ID فى الجدول . ويبين هذا البحث ، مع هيكل عبارة IF الخطية التى تحول REGION - ID الى دليل REGION - NUMBER ، فى جزء التركيم المراجع للمشكلة .

```
ACCUMULATE-SALES-TOTAL.
MOVE "NO" TO ERROR-SW
IF REGION-ID EQUAL "XYZ"
    MOVE 1 TO REGION-NUMBER
ELSE IF REGION-ID EQUAL "ABC"
    MOVE 2 TO REGION-NUMBER
ELSE IF REGION-ID EQUAL "GHF"
    MOVE 3 TO REGION-NUMBER
ELSE
    MOVE "YES" TO ERROR-SW

MOVE 1 TO DEPARTMENT-NUMBER
PERFORM LOOKUP-DEPARTMENT
    UNTIL ERROR-SW EQUAL "YES" OR
        DEPARTMENT-ID EQUAL
            VALID-DEPARTMENT-ID (DEPARTMENT-NUMBER)
IF SALES-AMOUNT NOT NUMERIC
    MOVE "YES" TO ERROR-SW
```

```

IF ERROR-SW EQUAL "YES"
    PERFORM INVALID-INPUT-ROUTINE.
ELSE
    ADD SALES-AMOUNT TO
        TOTAL-SALES-AMOUNT (REGION-NUMBER DEPARTMENT-NUMBER)

    PERFORM GET-SALES-RECORD.

.....
LOOKUP-DEPARTMENT.
IF DEPARTMENT-NUMBER EQUAL DEPARTMENT-LIMIT
    MOVE "YES" TO ERROR-SW
ELSE
    ADD 1 TO DEPARTMENT-NUMBER

```

في المشاكل الأكثر واقعية .. يجب ألا تكتب قيم ID - REGION ككوابيت ، وإنما تكتب كمناصر بيانات في مخزن العمل ، لها قيم VALUE المطلوبة ، أي إن :

```
05 REGION-ONE-ID    PIC X(3)    VALUE "XYZ".
```

إذا لم تميز ID - REGION .. يعد مفتاح الخطأ . وبالمثل .. إذا لم يوجد ID - DEPARTMENT في ID الموجودة في الجدول .. يعد نفس مفتاح الخطأ . ويعد مفتاح الخطأ كذلك إذا كان SALES - AMOUNT غير عددي .

التحميل بجزء VALUE

كما هو الحال مع الجداول ذات البعد الواحد (انظر القسم ٣ من هذا الفصل) .. فإنه من الممكن وضع قيم ابتدائية لجدول ذي بعدين ، وذلك بإعادة تعريف منطقة يتفق تكوينها مع الجدول مرة أخرى . لا يوصى بهذه الطريقة إذا كان من المتوقع أن تجرى تغييرات على الجدول خلال فترة استخدامه .

مثال ١٠ - ٢٠ :

فيما يلي وضع قيم ابتدائية لجدول مثال (١٠ - ١٢) .

```

01 FREIGHT-CHARGES-AREA.
   05 FREIGHT-CHARGE-VALUES.

      10 FILLER.
         15 FILLER      PIC S9(3)      COMP-3      VALUE +50.
         15 FILLER      PIC S9(3)V99   COMP-3      VALUE +1.50.
         15 FILLER      PIC S9(3)V99   COMP-3      VALUE +2.75.
         15 FILLER      PIC S9(3)V99   COMP-3      VALUE +2.25.
      10 FILLER.

```

```

15 FILLER PIC S9(3) COMP-3 VALUE +100.
15 FILLER PIC S9(3)V99 COMP-3 VALUE +3.00.
15 FILLER PIC S9(3)V99 COMP-3 VALUE +4.23.
15 FILLER PIC S9(3)V99 COMP-3 VALUE +4.16.
10 FILLER.
15 FILLER PIC S9(3) COMP-3 VALUE +999.
15 FILLER PIC S9(3)V99 COMP-3 VALUE +3.50.
15 FILLER PIC S9(3)V99 COMP-3 VALUE +5.45.
15 FILLER PIC S9(3)V99 COMP-3 VALUE +5.20.
05 FREIGHT-CHARGES-TABLE REDEFINES FREIGHT-CHARGE-VALUES.
10 WEIGHT-RANGE OCCURS 3 TIMES.
15 WEIGHT-LIMIT PIC S9(3) COMP-3.
15 DESTINATION OCCURS 3 TIMES.
20 FREIGHT-CHARGE PIC S9(3)V99 COMP-3.

```

التحميل بعبارة READ

هذا الإجراء - بتمييزاته - متماثل مع الجدول ذي البعد الواحد (انظر القسم ١٠ - ٢) .

مثال ١٠ - ٢١ :

يحمل جدول مثال (١٠ - ١٢) من ملف المدخلات :

```

FD TABLE-VALUES-FILE...
01 TABLE-VALUES-RECORD.
05 WEIGHT-VALUE PIC S9(3) COMP-3.
05 FREIGHT-VALUE PIC S9(3)V99 COMP-3.
OCCURS 3 TIMES.

```

أى إن كل سجل TABLE - VALUES - RECORD يوضع فى عنصر WEIGHT - RANGE ، بحيث يوضع WEIGHT - VALUE فى WEIGHT - LIMIT ، وتوضع القيم لثلاث من FREIGHT - VALUE فى المواقع الثلاث FREIGHT - CHARGES .
وتؤدى الشفرة التالية العمل .

```

.....
OPEN INPUT TABLE-VALUES-FILE
MOVE "NO" TO END-FILE-SW
MOVE 1 TO NUMBER-WEIGHT-ENTRIES
PERFORM GET-TABLE-RECORD
PERFORM LOAD-TABLE
UNTIL END-FILE-SW EQUAL "YES" OR NUMBER-WEIGHT-ENTRIES
GREATER THAN SIZE-OF-WEIGHT-TABLE
IF END-FILE-SW EQUAL "NO"
DISPLAY "FREIGHT-CHARGES-TABLE TOO SMALL: CHANGE PROGRAM"
UPON OPERATOR-MESSAGE-DEVICE
SUBTRACT 1 FROM NUMBER-WEIGHT-ENTRIES
CLOSE TABLE-VALUES-FILE
.....

```


GET-TABLE-RECORD.

READ TABLE-VALUES-FILE

AT END

MOVE "YES" TO END-FILE-SW

LOAD-TABLE.

MOVE WEIGHT-VALUE TO WEIGHT-LIMIT (NUMBER-WEIGHT-ENTRIES)

PERFORM MOVE-FREIGHT-CHARGES

VARYING NUMBER-FREIGHT-CHARGES FROM 1 BY 1

UNTIL NUMBER-FREIGHT-CHARGES GREATER THAN

SIZE-OF-FREIGHT-TABLE

PERFORM GET-TABLE-RECORD

ADD 1 TO NUMBER-WEIGHT-ENTRIES

MOVE-FREIGHT-CHARGES.

MOVE FREIGHT-VALUE (NUMBER-FREIGHT-CHARGES) TO

FREIGHT-CHARGE (NUMBER-WEIGHT-ENTRIES NUMBER-FREIGHT-CHARGES)

والمعرف الدليلان وحدودهما في قسم مخزن العمل :

01 TABLE-DIMENSIONS.

05 NUMBER-WEIGHT-ENTRIES PIC S9(5) COMP SYNC.

05 NUMBER-FREIGHT-CHARGES PIC S9(5) COMP SYNC.

05 SIZE-OF-WEIGHT-TABLE PIC S9(5) COMP SYNC VALUE +3.

05 SIZE-OF-FREIGHT-TABLE PIC S9(5) COMP SYNC VALUE +3.

لاحظ أنه بذل كل جهد لجعل البرنامج مرناً وسهلاً في تغييره بقدر الإمكان . وعلى هذا .. تم حساب عدد عناصر الأوزان النشطة (كعدد سجلات منطقية في حلف مدخلات) ، بحيث يمكن أن يضبط البرنامج نفسه بطريقة مناسبة بين 1 و 3 لاحتويات WEIGHT - RANGE . والأكثر هو ظهور SIZE - OF - FREIGHT - TABLE و SIZE - OF - WEIGHT - TABLE في الشفرة كعناصر بيانات ، بحيث إنهما إذا كان حجم الجدول مطلوب تغييره .. فلن يحدث تعديلاً إلا في قيم OC - VALUE ، CURS المناسبة فقط . (انظر المسألة ١٠ - ٧٨) .

١ - ٦ جداول متغيرة الطول :

جزء الحدوث طبقاً لـ ...

حتى الآن تعرضنا لجدول لها عدد ثابت من العناصر ، ويخصص لها عادة مكان يكفي أقصى عدد متوقع لعناصرها ، مع استخدام عداد counter لعدد العدد الفعلي للعناصر النشطة active ، ولا توجد في الكويل طريقة لحفظ الذاكرة الرئيسية ، عندما يكون الجدول مملوواً بالكامل . ولكن عندما يتم إخراج الجداول على قرص أو شريط .. يقدم الكويل آلية تجعل العناصر النشطة فقط جزءاً من السجل المنطقي ، محافظة بذلك على مساحة القرص أو الشريط :

مثال ١٠ - ٢٢ :

اعتبر الملف التالي :

```

FD PRODUCT-MASTER-FILE ...
01 PRODUCT-MASTER-RECORD.
    05 PRODUCT-ID          PIC X(5).
    05 PRODUCT-DESCRIPTION PIC X(20).
    05 NUMBER-OF-SUPPLIERS PIC S9(2) COMP.
    05 SUPPLIER-INFORMATION OCCURS 1 TO 50 TIMES
                           DEPENDING ON NUMBER-OF-SUPPLIERS.
        10 SUPPLIER-ID      PIC X(4).
        10 SUPPLIER-NAME    PIC X(20).
        10 SUPPLIER-PRICE   PIC S9(3)V99 COMP-3.
        10 SUPPLIER-TERMS   PIC X(2).
    
```

يحدد جزء OCCURS ... DEPENDING للمتجم أنه بالرغم من أن أقصى حجم الجدول هو 50 عنصراً ، إلا أن الحجم النشط active للجدول (الحجم الفعال على القرص أو الشريط) يتحدد عن طريق محتويات NUMBER - OF - SUPPLIERS ، والذي يمكن أن يتراوح من واحد إلى خمسين .

عندما يكون جدول متغير الطول variable - length table (كما هو معرف بجزء OCCURS ... DEPENDING ..) جزءاً من سجل منطقي ، يحتوى الملف على سجلات متغيرة الطول (الفصل الخامس) . تكوين ... OCCURS .. DEPENDING لجدول متغيرة الطول ، هو ما يلي :

OCCURS integer-1 TO integer-2 TIMES
DEPENDING ON data-name

حيث integer - 1 يجب أن يكون 1 على الأقل ، وأقل من integer - 2 - data - name. integer (له أى شكل من أشكال USAGE) والذي يمكن - أثناء التنفيذ - أن يفترض قيمة صحيحة تقع بين integer - 1 ، integer - 2 بما فى ذلك الحدود . والقيمة الحالية لاسم البيانات data - name التى تحدد عدد عناصر الجدول النشطة حالياً (وبالتالى تحدد المكان اللازم لسجل منطقي متغير الطول فى التخزين المساعد) . لاحظ أنه يجب أن يكون فى الجدول متغير الطول عنصر واحد على الأقل (انظر ملحق د بالنسبة لكويل 1985 النمطى) .

عندما يوجد جدول متغير الطول فى سجل ... فالمطلوب أن يكون هو آخر عنصر فى السجل ؛ فإذا كانت الجداول متداخلة .. فإن الجدول متغير الطول هو جدولاً أعلى مستوى فقط . وفى كلمات أخرى .. لا يمكن أن يكون جزء DE- OCCURS PENDING تابعاً لى جزء OCCURS آخر .

نقل جداول متغيرة الطول

إذا كان من المرغوب فيه نقل جدول متغير الطول من منطقة بيانات إلى منطقة أخرى ... فيجب أن توضع قيم عنصرى DE- PENDING متساوية ، قبل تنفيذ MOVE . (انظر ملحق د بالنسبة الى تعديلات كويل 1985 النمطى) .

مثال ١٠ - ٢٣ :

تحتوى الطريقة الصحيحة لوضع سجل منطقي - فى مخزن العمل - على جدول متغير الطول :

```

FD ASSEMBLY-FILE...
01 ASSEMBLY-RECORD.
   05 ITEM-ID PIC X(6).
   05 NUMBER-SUBASSEMBLIES PIC S9(2) COMP.
   05 SUBASSEMBLY-INFO OCCURS 1 TO 25 TIMES
                        DEPENDING ON
                        NUMBER-SUBASSEMBLIES.
       10 SUBPART-ID PIC X(6).
       10 WAREHOUSE-LOCATION PIC XX.
       10 QUANTITY-AVAILABLE PIC S9(5) COMP-3.
.....
WORKING-STORAGE SECTION.
01 WS-ASSEMBLY-RECORD.
   05 WS-ITEM-ID PIC X(6).
   05 WS-NUMBER-SUBASSEMBLIES PIC S9(2) COMP.
   05 WS-SUBASSEMBLY-INFO OCCURS 1 TO 25 TIMES
                        DEPENDING ON
                        WS-NUMBER-SUBASSEMBLIES.
       10 WS-SUBPART-ID PIC X(6).
       10 WS-WAREHOUSE-LOCATION PIC XX.
       10 WS-QUANTITY-AVAILABLE PIC S9(5) COMP-3.
.....
READ ASSEMBLY-FILE AT END...
MOVE NUMBER-SUBASSEMBLIES TO WS-NUMBER-SUBASSEMBLIES
MOVE ASSEMBLY-RECORD TO WS-ASSEMBLY-RECORD

```

لاحظ أن عبارة READ ASSEMBLY - FILE INTO WS - ASSEMBLY - RECORD بسيطة لا تؤدي العمل ؛
حيث إنها تكافئ :

```

READ ASSEMBLY-FILE...
MOVE ASSEMBLY-RECORD TO WS-ASSEMBLY-RECORD

```

والتي لا تساوي WS - NUMBER - SUBASSEMBLIES ، مع NUMBER - SUBASSEMBLIES ، وذلك قبل تنفيذ عبارة النقل . ويمكن أن تكون نتيجة النقل حذفاً أو إضافة في WS - ASSEMBLY - RECORD طبقاً للمحتويات السابقة في WS - NUMBER - SUBASSEMBLIES .

توسيع أو تضيق الجداول

يمكن إضافة أو حذف عناصر نشطة إلى ومن جدول متغير الطول ؛ بشرط عدم تعدى حدود TIMES في جزء OCCURS . فإذا كان مطلوباً إضافة أحد العناصر .. فيجب أن تزداد قيمة عنصر DEPENDING بمقدار 1 .

يجب ألا تجدى التغييرات في جدول متغير الطول التي تؤثر على حجم الجدول في الذاكرة الاحتياطية لإدخال السجل المنطقي (حيث إنها يمكن أن تثير أى سجلات منطقية تالية في الملف المستخدم للتجميع) . ويمكن إجراء مثل هذه التغييرات في نسخة مخزن العمل لسجل المدخلات ، أو في الذاكرة الاحتياطية لسجل المخرجات .

مثال ١٠ - ٢٤ :

لإضافة معلومات من :

```
FD TRANSACTION-FILE ...
01 TRANSACTION-RECORD.
   05 INPUT-ITEM-ID          PIC X(6).
   05 INPUT-SUBPART-ID       PIC X(6).
   05 INPUT-WAREHOUSE-LOCATION PIC XX.
   05 INPUT-QUANTITY-AVAILABLE PIC S9(5).
```

إلى WS - ASSEMBLY - RECORD في مثال (١٠ - ٢٣) .. فإننا نكتب الشفرة التالية :

```
ADD 1 TO WS-NUMBER-SUBASSEMBLIES
MOVE INPUT-SUBPART-ID TO
      WS-SUBPART-ID (WS-NUMBER-SUBASSEMBLIES)
MOVE INPUT-WAREHOUSE-LOCATION TO
      WS-WAREHOUSE-LOCATION (WS-NUMBER-SUBASSEMBLIES)
MOVE INPUT-QUANTITY-AVAILABLE TO
      WS-QUANTITY-AVAILABLE (WS-NUMBER-SUBASSEMBLIES)
```

لاحظ أن عنصر DEPENDING ازداد بمقدار 1 أولاً ؛ مضافاً إلى ذلك أن الجدول يتسع للعنصر الجديد . وعلى ذلك يخدم عنصر DEPENDENT كدليل لتوقيع العنصر الجديد في نهاية الجدول (أسهل موقع لوضعه انظر مثال (١٠ - ٢٧) لمعرفة طريقة أخرى) .

مثال ١٠ - ٢٥ :

افترض أن INPUT - SUBPART - ID يعرف جزءاً مجتمعاً تجميعاً جزئياً ، مطلوب حذفه من - WS - ASSEMBLY - RECORD في مثال (١٠ - ٢٣) .

(١) في الحذف المنطقي logical deletion .. ننقل فراغات (أو أى رمز حذف deletion code آخر) الى عنصر الجدول الذى يتفق مع INPUT - SUBPART - ID ، وتحدد الفراغات أن العنصر غير نشيط لاي برنامج ، يجرى تشغيلاً على الجدول. إلا أننا لم نقل WS - NUMBER - SUBASSEMBLIES ؛ حيث إن العنصر لم يحذف واقعياً physically .

```
MOVE "YES" TO ENTRY-IN-TABLE
MOVE 1 TO TABLE-SUBSCRIPT
PERFORM LOCATE-SUBPART
      UNTIL ENTRY-IN-TABLE EQUAL "NO"
      OR INPUT-SUBPART-ID EQUAL
        WS-SUBPART-ID (TABLE-SUBSCRIPT)
```

```

IF ENTRY-IN-TABLE EQUAL "NO"
  PERFORM ERROR-ROUTINE
ELSE
  MOVE SPACES TO WS-SUBPART-ID (TABLE-SUBSCRIPT)
  .....
LOCATE-SUBPART.
  IF TABLE-SUBSCRIPT EQUAL WS-NUMBER-SUBASSEMBLIES
    MOVE "NO" TO ENTRY-IN-TABLE
  ELSE
    ADD 1 TO TABLE-SUBSCRIPT

```

بوجود فراغات في (j) WS - SUBPART - ID .. فان : أى برنامج مستقبلي يجرى تشغيلاً على الجدول ، يهمل كذلك (j) WS - QUANTITY - AVAILABLE و (j) WS - WAREHOUSE - LOCATION .

(ب) في الحذف الواقعي physical deletion .. ننقل كل العناصر التي تتبع العنصر المتفق مع INPUT - SUBPART - ID موقعا واحدا (في اتجاه بداية الجدول) ، ونقل قيمة عنصر DEPENDING بمقدار واحد ، حيث حذف عنصر واحد حذفاً واقعياً .

```

MOVE "YES" TO ENTRY-IN-TABLE
MOVE 1 TO TABLE-SUBSCRIPT
PERFORM LOCATE-SUBPART
  UNTIL ENTRY-IN-TABLE EQUAL "NO"
  OR INPUT-SUBPART-ID EQUAL
    WS-SUBPART-ID (TABLE-SUBSCRIPT)
IF ENTRY-IN-TABLE EQUAL "NO"
  PERFORM ERROR-ROUTINE
ELSE
  PERFORM MOVE-UP-TABLE-ENTRIES
    VARYING MOVING-SUBSCRIPT FROM TABLE-SUBSCRIPT BY 1
    UNTIL MOVING-SUBSCRIPT EQUAL WS-NUMBER-SUBASSEMBLIES
  SUBTRACT 1 FROM WS-NUMBER-SUBASSEMBLIES
  .....
LOCATE-SUBPART.
  IF TABLE-SUBSCRIPT EQUAL WS-NUMBER-SUBASSEMBLIES
    MOVE "NO" TO ENTRY-IN-TABLE
  ELSE
    ADD 1 TO TABLE-SUBSCRIPT
MOVE-UP-TABLE-ENTRIES.
  ADD 1 MOVING-SUBSCRIPT GIVING MOVING-SUBSCRIPT-PLUS-1
  MOVE WS-SUBASSEMBLY-INFO (MOVING-SUBSCRIPT-PLUS-1) TO
    WS-SUBASSEMBLY-INFO (MOVING-SUBSCRIPT)

```

لاحظ كيف وضعت SUBPART UNTIL - PERFORM LOCATE الدليل TABLE - SUBSCRIPT في رقم الموقع الخاص بالعنصر الذي سيحذف إذا كان العنصر موجودا في الجدول. يجب أن تنقل كل العناصر التي تقع بين - TABLE SUBSCRIPT ونهاية الجدول (في WS - NUMBER - SUBASSEMBLIES) موقعا واحدا ، لضغط مكان العنصر المحذوف من الجدول . ويحدث ذلك في MOVE - UP - TABLE - ENTRIES ؛ حيث ينقل العنصر عند MOVING - SUBSCRIPT +1 إلى العنصر عند MOVING - SUBSCRIPT . يتغير MOVING - SUBSCRIPT من - TABLE SUBSCRIPT إلى 1 - WS - NUMBER - SUBASSEMBLIES (تذكر أنه عند استخدام EQUAL بدلا من GREATER في عبارة ... UNTIL ... VARYING ... PERFORM .. فإن آخر قيمة لعنصر VARYING ، والتي ينفذ لها المقطع فعلا تقل "1" عن القيمة المحددة للتوقف) و هذا ضروري بسبب أن مواقع العناصر التي تنقل تحددت بواسطة MOV - SUBSCRIPT مضافا « ١ » إليها . وحيث إن الدليل لا يمكن أن يكون تعبيراً .. فمن الضروري تعريف عنصر بيانات إضافي 1 - MOVING - SUBSCRIPT - PLUS ، يحسب فيه 1 + MOVING - SUBSCRIPT . بعد نقل كل العناصر المناسبة موقعا واحدا .. تقل قيمة عنصر DEPENDING بمقدار « 1 » لتحديد أن عناصر الجدول قد قلت الآن . يفضل الحذف الواقعي عن الحذف المنطقي للجداول متغيرة الطول ؛ لأنه يقلل حجم مواقع التخزين المساعد اللازمة لحفظ الجدول .

١ - ٧ الفهارس

عندما تستخدم الأدلة في تعريف عناصر الجدول .. يجب أن ينتج المترجم تعليمات لغة آلة لتحويل قيمة الدليل إلى العنوان الفعلي للذاكرة وذلك للعنصر المطلوب . الفهارس indexes هي أسماء بيانات كويل ، يمكن استخدامها بدلا من الدلائل في جعل حسابات عناوين لغة الآلة أكثر كفاءة (انظر المسائلين ٧٤ و ٧٥ من هذا الفصل) . وقيمة الفهرس هي إزاحة displacement في الجدول ؛ أي إنها رقم عندما يضاف إلى عنوان بداية الجدول .. ينتج عنه عنوان العنصر المطلوب . الفهارس أكثر كفاءة من الدلائل ، ويجب استخدامها كلما أمكن .

تعرف الفهارس المستخدمة بدلا من الدلائل لأحد الجداول في جزء OCCURS الذي يعرف الجدول ، ولا تحتاج أي تعريف خاص بها (على عكس الدلائل) ، ويبين تكوين جزء OCCURS عند تعريف فهرس في شكل (١٠ - ٢) .

```
OCCURS integer-1 [TO integer-2] TIMES
  [DEPENDING ON data-name-1]
  [INDEXED BY index-name-1 [index-name-2] ...]
```

شكل (١٠ - ٢)

أصبحت 1 - index - name ، 2 - index - name و ... إلخ ، متاحة لاستخدامها بدلا من الدلائل عند الاتصال بمحتويات الجدول .

مثال ١٠ - ٢٦ :

```

01 SALES-FIGURES-BY-STATE.
05 STATE-SALES OCCURS 50 TIMES
INDEXED BY STATE-NUMBER.
10 STATE-ID PIC X(2).
10 SALES-AMOUNT PIC S9(5)V99 COMP-3.

```

هنا تم تعريف جدول مثال (١٠ - ٤) باستخدام الفهارس . يتاح الفهرس STATE - NUMBER الآن لاستخدامه بدلا من STATE - SALES - SUB . لاحظ أن الفهرس يصاحبه جدول خاص particular ، ويجب ألا يستخدم STATE - NUMBER في الاتصال بأي جدول آخر، إلا إذا كان لهذا الجدول نفس الهيكل تماما (USAGE , PICTURE) ، والمستويات وعدد المحتويات ، وغيرها (مثل STATE - SALES . لاحظ كذلك أنه لا توجد حاجة لتعريف أكثر للفهرس (غير مسموح بذلك أيضا)

لا يمكن معاملة الفهارس مثل الدلائل : إذ يمكن وضع قيمة ابتدائية للفهرس ، أو تغيير الفهرس (١) باستخدام عبارة SET فقط (٢) ، أو عبارة PERFORM (٣) ، أو عبارة SEARCH . (انظر القسم ١٠ - ٨) . يمكن استخدام عبارة SET في وضع قيمة ابتدائية لفهرس ، سواء كانت قيمة فهرس آخر ، أو عنصر بيانات عددياً ، أو ثابتاً عددياً . وعكسياً .. يمكن استخدام عبارة SET في وضع عنصر بيانات عددي في القيمة الحالية لأحد الفهارس ، وتكوينها هو كما يلي :

$$\underline{\text{SET}} \left\{ \begin{array}{l} \text{index-name-1} \\ \text{data-name-1} \end{array} \right\} \left[\begin{array}{l} \text{index-name-2} \\ \text{data-name-2} \end{array} \right] \dots \underline{\text{TO}} \left\{ \begin{array}{l} \text{index-name-3} \\ \text{identifier-3} \\ \text{literal} \end{array} \right\}$$

مثال ١٠ - ٢٧ :

- SET STATE-NUMBER TO 3

تحويل عبارة SET القيمة 3 إلى إزاحة العنصر الثالث في STATE - SALES (انظر مثال ١٠ - ٢٦) .

- SET STATE-NUMBER TO SOME-OTHER-INDEX

- SET STATE-NUMBER TO A-SUBSCRIPT

● SET A-SUBSCRIPT TO STATE-NUMBER

$$\text{SET index-name-1 [index-name-2] ... } \left\{ \begin{array}{l} \text{UP BY} \\ \text{DOWN BY} \end{array} \right\} \left\{ \begin{array}{l} \text{data-name} \\ \text{literal} \end{array} \right\}$$

مثال ۱۰ - ۲۸ :

- SET STATE-NUMBER DOWN BY 1

● SET STATE-NUMBER UP BY A-SUBSCRIPT

تزداد قيمة STATE - NUMBER بمقدار الإزاحة التي تجعله يشير إلى أن عدد العناصر يساوى المحتوى الحالى - A - SUBSCRIPT (مثل إذا بدأ STATE - NUMBER في الإشاره إلى ثالث عنصر ، واحتوى A - SUBSCRIPT على القيمة 5 .. فعند ذلك يشير STATE - NUMBER إلى العنصر الثامن) .

يمكن استخدام الفهارس في عبارات PERFORM .. وفي شروط IF كمعاصر بيانات عديدة صحيحة تماما .

مثال ١٠ - ٢٩ :

أوجد إجمالى قيمة المبيعات في مثال (١٠ - ٢٦) .

```
01 SALES-FIGURES-BY-STATE.
   05 STATE-SALES OCCURS 50 TIMES INDEXED BY STATE-NUMBER.
      10 STATE-ID PIC X(2).
      10 SALES-AMOUNT PIC S9(5)V99 COMP-3.
```

```
.....
MOVE ZERO TO TOTAL-SALES
PERFORM ADD-UP-STATE-SALES
    VARYING STATE-NUMBER FROM 1 BY 1
    UNTIL STATE-NUMBER GREATER THAN 50
.....
```

```
ADD-UP-STATE-SALES.
    ADD SALES-AMOUNT (STATE-NUMBER) TO TOTAL-SALES
```

لاحظ أن STATE - NUMBER يستخدم مثل الدليل تماماً . وفي الواقع .. بالنظر فقط إلى جزء OCCURS ، يمكن القول بأنه ليس دليلاً . وبالرغم من ذلك .. تنتج عن الفهرس STATE - NUMBER لغة آلة أكثر كفاءة .

مثال ١٠ - ٢٠ :

أعد كتابة LOOK-UP - STATUS من مثال (١٠ - ١٠) ؛ مستخدماً فهرساً بدلاً من الدليل .

```
LOOK-UP-STATUS.
    SET STATUS-TABLE-INDEX TO 1
    MOVE "YES" TO VALID-STATUS-SW
    PERFORM CHECK-ENTRY.
        UNTIL VALID-STATUS-SW EQUAL "NO"
        OR STATUS-CODE (STATUS-TABLE-INDEX) EQUAL INPUT-STATUS-CODE
CHECK-ENTRY.
    IF STATUS-TABLE-INDEX EQUAL 40
        MOVE "NO" TO VALID-STATUS-SW
    ELSE
        SET STATUS-TABLE-INDEX UP BY 1
```

يجب وضع SET الفهرس ، وإلا فإنه يصبح دليلاً .

مثال ١٠ - ٢١ :

اعد كتابة مثال (١٠ - ١٨) : مستخدما فهرساً بدلاً من الدليل .

```
FD SALES-INPUT-FILE...
.....
01 TOTAL-SALES-AMOUNTS-TABLE.
   05 REGION OCCURS 7 TIMES
               INDEXED BY REGION-INDEX.
      10 DEPARTMENT OCCURS 12 TIMES
                  INDEXED BY DEPARTMENT-INDEX.
         15 TOTAL-SALES-AMOUNT PIC S9(5)V99 COMP.
.....
ELSE
  SET REGION-INDEX TO REGION-ID
  SET DEPARTMENT-INDEX TO DEPARTMENT-ID
  ADD SALES-AMOUNT TO TOTAL-SALES (REGION-INDEX DEPARTMENT-INDEX)

PERFORM GET-SALES-RECORD
```

الفهرسة النسبية

يمكن جمع أو طرح ثابت عددي (وثوابت عددية فقط) من الفهرس للإشارة إلى عنصر بالنسبة إلى القيمة الحالية للفهرس .
وعادة ما يكون هذا مقنعا ، عند نقل عناصر الجدول هنا وهناك داخل الجدول .

مثال ١٠ - ٢٢ :

أفرض أنه يرغب في نقل عنصر من الموقع K إلى الموقع K-5 . باستخدام الأداة يمكننا أن نعرف.

```
01 TABLE-SUBSCRIPTS.
   05 K PIC S9(5) COMP SYNC.
   05 K-MINUS-FIVE PIC S9(5) COMP SYNC.
.....
  SUBTRACT 5 FROM K GIVING K-MINUS-FIVE
  MOVE TABLE-ENTRY (K) TO TABLE-ENTRY (K-MINUS-FIVE)
```

ونستطيع بالفهرسة استخدام جزء أكثر سهولة :

```
SET TABLE-INDEX TO K
MOVE TABLE-ENTRY (TABLE-INDEX) TO TABLE-ENTRY (TABLE-INDEX - 5)
```

١٠ - ٨ البحث المتتالي فى الجداول وفعل «ابحث»

سبق أن أوضحت الأقسام السابقة كيفية برمجة البحث فى جدول باستخدام بورة PERFORM . إلا أن كويل ANS النمطى يقوم فعل بحث SEARCH خاص ، أسهل فى استخدامه وأكثر كفاءة كذلك ؛ فإذا احتوى الجدول على n عنصر .. فإن البحث المتتالى (الخطى) يفحص العناصر بالترتيب K ثم K+1 ثم ... K+2 ثم n-1 ثم n . (عادة ما تكون مساوية 1 (لعبارة البحث المتتالى التكوين الموجود فى شكل (١٠ - ٣) ؛ يجب أن يكون identifier هو اسم الجدول المراد البحث فيه (أى إنه عنصر بيانات موصوف باستخدام OCCURS) . يحدد جزء AT END المطلوب عمله ، عندما يفشل البحث (أى عندما لا يتحقق أى شرط من شروط WHEN أثناء البحث) .

```
SEARCH identifier-1 [ VARYING { identifier-2
                             { index-name-1 } } ]
[ AT END imperative-statement(s)-1
  WHEN condition-1 { imperative-statement(s)-2
                    { NEXT SENTENCE
[ WHEN condition-2 { imperative-statement(s)-3
                    { NEXT SENTENCE
                    } ] ] ] ] ...
```

شكل (١٠ - ٣)

مثال ١٠ - ٢٢ :

أعد مثال (١٠ - ١٠) مستخدماً أمر البحث بدلاً من أمر التنفيذ .

```
01 VALID-STATUS-CODES-TABLE.
   05 STATUS-CODE PIC X(3)
                      OCCURS 40 TIMES
                      INDEXED BY VALID-CODE-INDEX.
.....
LOOK-UP-STATUS.
  SET VALID-CODE-INDEX TO 1
  SEARCH STATUS-CODE
    AT END
      MOVE "NO" TO INPUT-CODE-OK-SW
    WHEN INPUT-STATUS-CODE EQUAL STATUS-CODE (VALID-CODE-INDEX)
      MOVE "YES" TO INPUT-CODE-OK-SW
```

لاحظ أنه يجب تعريف الجدول بفهرس index ، إذا ما كان فعل SEARCH مستخدماً . كما يجب أن يضع المبرمج فهرس البحث عند العنصر الذى يراد للبحث أن يبدأ منه (عادة ما يكون «1») . وتعمل عبارة SEARCH السابقة كما يلى :

(١) نظراً لأن جزء VARYING غير محدد .. تستخدم SEARCH أول فهرس معرف لجدول STATUS - CODE . وفى هذه الحالة يكون أول فهرس (والأول فقط) VALID - CODE - INDEX .

(٢) تقارن قيمة فهرس SEARCH مع عدد عناصر الجلول (هنا 40) . إذا زادت قيمة الفهرس عن حجم الجلول .. ينفذ عبارة AT END ، ويستمر - عند ذلك - التنفيذ بأول عبارة بعد النقطة التي تنهى عبارة SEARCH .

(٣) إذا لم يتعد الفهرس حجم الجدول .. يتم تقويم شروط WHEN (بالترتيب الذى كتبت به) . ويتسبب أول شرط WHEN متحقق فى تنفيذ العبارة المصاحبة له . وعندها .. ينتهى البحث (ويستمر التنفيذ بأول عبارة تتبع النقطة التي تنهى عبارة البحث) .

(٤) إذا لم يتحقق أى شرط من شروط WHEN .. يزداد فهرس SEARCH بمقدار عنصر واحد ، ويعود تنفيذ عبارة البحث إلى الخطوة رقم (٢) المذكورة أعلاه .

مثال ١٠ - ٣٤ :

باستخدام SEARCH ... VARYING .. نستطيع فحص جداول متوازية parallel tables ، وهى جداول تحتوى عناصرها المتناظرة على بيانات عن نفس العنصر . افرض أن لدينا الجداول المتوازية التالية :

01 DEDUCTION-CODES-TABLE.	
05 DEDUCT-CODE	PIC X OCCURS 30 TIMES INDEXED BY CODE-INDEX.
01 DEDUCTIONS-INFO-TABLE.	
05 PAYROLL-DEDUCTIONS	OCCURS 30 TIMES INDEXED BY INFO-INDEX.
10 DEDUCT-DESCRIPTION	PIC X(25).
10 DEDUCT-AMOUNT	PIC \$99V99 COMP-3.

حيث (k) DEDUCT - CODE - يناظر (k) PAYROLL - DEDUCTIONS . بمعرفة أحد المدخلات - PAYROLL - CODE .. فإننا نقوم بتشغيل خصومات الراتب باستخدام SEARCH ... VARYING :

```

SET CODE-INDEX TO 1
SET INFO-INDEX TO 1
SEARCH DEDUCT-CODE
  VARYING INFO-INDEX
  AT END
    MOVE "YES" TO PAYROLL-ERROR-SW
  WHEN DEDUCT-CODE (CODE-INDEX) EQUAL PAYROLL-CODE
    MOVE DEDUCT-DESCRIPTION (INFO-INDEX) TO OUTPUT-AREA
    SUBTRACT DEDUCT-AMOUNT (INFO-INDEX) FROM NET-PAY
    
```

حيث إن الجدولين لهما هيكلان مختلفان .. فمن الضروري استخدام فهرسين . وعندما يحدد جزء VARYING فهرساً ينتمى إلى جدول غير جدول SEARCH (١) تزداد قيمة أول فهرس محدد فى جدول SEARCH (وهو CODE - INDEX) اثناء البحث . (٢) كلما ازدادت قيمة أول فهرس لجدول SEARCH ... إزدادت كذلك قيمة فهرس VARYING (هو - INFO INDEX) . وعلى هذا فإن فهرس VARYING يتبع فهرس SEARCH ؛ بحيث يشيران دائماً إلى المحتويات المتناظرة مع بعضها البعض ، بافتراض أنها وضعت لـ "SET نفس القيمة قبل تنفيذ عبارة البحث .

إذا كان للجدول أكثر من فهرس واحد معرف له ، وتسمى VARYING فهرساً غير الفهرس الأول ؛ فيستخدم هذا الفهرس فى البحث .

حيث إن SEARCH ترتبط مع جزء OCCURS واحد فى نفس الوقت .. فلا بد من كتابة شفرة خاصة لإجراء بحث كامل فى جدول ذى بعددين . وعادة ما يكفى وضع عبارة بحث فى مقطع PERFORM ، مع ترك عبارة PERFORM .. تغير VARY أول فهرس (ويتغير الثانى بواسطة عبارة البحث) .

مثال ١٠ - ٢٥ :

لتحديد (5) INPUT - COURSE PIC.X فى الجدول :

01 STUDENT-TRANSCRIPT.	
05 STUDENT-ID	PIC X(9).
05 SEMESTERS-ATTENDED	PIC S9 COMP.
05 SEMESTER-INFORMATION	OCCURS 1 TO 8 TIMES
	DEPENDING ON SEMESTERS-ATTENDED
	INDEXED BY SEMESTER-INDEX.
10 SEMESTER-DATE	PIC X(6).
10 NUMBER-COURSES	PIC S9 COMP.
10 COURSE-INFO	OCCURS 6 TIMES
	INDEXED BY COURSE-INDEX.
15 COURSE-ID	PIC X(5).
15 CREDITS	PIC S9V9 COMP-3.
15 GRADE	PIC S9 COMP-3.

ونكتب الشفرة على النحو التالى :

```

MOVE "NO" TO FOUND-IT-SW
PERFORM LOCATE-COURSE
  VARYING SEMESTER-INDEX FROM 1 BY 1
  UNTIL SEMESTER-INDEX GREATER THAN SEMESTERS-ATTENDED
  OR FOUND-IT-SW EQUAL "YES"
IF FOUND-IT-SW EQUAL "YES"
  SET SEMESTER-INDEX DOWN BY 1
  PERFORM PROCESS-COURSE-INFO

```

```
ELSE
  PERFORM COURSE-NOT-FOUND-ERROR
```

```
LOCATE-COURSE.
  SET COURSE-INDEX TO 1
  SEARCH COURSE-INFO
    WHEN COURSE-INDEX GREATER THAN
      NUMBER-COURSES (SEMESTER-INDEX)
      NEXT SENTENCE
    WHEN COURSE-ID (SEMESTER-INDEX COURSE-INDEX)
      EQUAL INPUT-COURSE
      MOVE "YES" TO FOUND-IT-SW
```

تعليقات :

- (١) تنفذ عبارة البحث مرات ومرات مع تغيير SEMESTSR - INDEX من «1» الى SEMESTERS - ATTENDED ، وهو مداه الكامل من القيم الممكنة .
- (٢) توضع القيمة «1» للفهرس COURSE - INDEX فى كل مرة ينفذ LOCATE - COURSE . وعلى هذا .. يبحث فى محتويات الجدول COURSE - INFO ، لكل قيمة من قيم SEMESTER - INDEX .
- (٣) يفترق اسم جدول البحث COURSE - INFO إلى وجود أدلة أو فهارس فى عبارة البحث ، وذلك بالرغم من أنه محتوى فى جدول SEMESTER - INFORMATION ، وعادة ما يكون له أدلة أو فهارس .
- (٤) لا يعد عدم وجود جزء AT END مشأله هنا ؛ فإذا تم الوصول إلى نهاية COURSE - INFO لقيمة معينة من قيم SEMESTER - INDEX .. فسوف يستمر التنفيذ بالعبارة التى تلى عبارة البحث ، وهى - فى هذه الحالة - نهاية المقطع . إلا أن الوصول إلى نهاية LOCATE - COURSE يتسبب فى زيادة عبارة PERFORM من SEMESTER - INDEX ، وتنفذ المقطع مرة أخرى ؛ أى بحثاً عن COURSE - INFO للفصل الدراسى التالى ، كما هو مطلوب .
- (٥) يختبر جزء WHEN الأول النهاية المنطقية logical end لجدول COURSE - INFO ؛ فإذا كان هذا الشرط صحيحاً .. تتسبب NEXT SENTENCE فى استمرار التنفيذ حتى نهاية المقطع ، مع زيادة قيمة SEMESTER - INDEX ، كما فى التعليق رقم (٤) .
- (٦) إذا تحقق ثانى شرط WHEN .. توضع قيمة YES للمفتاح FOUND - IT - SW ؛ إلا أن PERFORM ... VARY- UNTIL (FOUND - IT - SW EQUAL شرط تقويماً لشرط "YES")

```
SET SEMESTER-INDEX DOWN BY 1
```

لا تجر الزيادة الإضافية .

١٠ - ٩ البحث الثنائى فى الجداول، وفعل ابحث الكل

خوارزمية البحث الثنائى binary search أكثر كفاءة من البحث التتابعى للجداول التى تكون كبيرة نسبياً ، والتى تكون مرتبة sorted ترتيباً تصاعدياً ascending sequence ، أو ترتيباً تنازلياً descending sequence ؛ طبقاً لبعض حقول الجدول ، والذي يسمى حقلاً رئيسياً ، أو مفتاح الترتيب key . وكلما ازداد حجم الجدول .. ازدادت أهمية البحث الثنائى عن البحث المتتابع . ويبدأ البحث الثنائى بفحص عنصر الوسط (يفسر بطريقة مناسبة إذا كان العدد زوجياً) . إذا كان العنصر المطلوب يقع فى منتصف الجدول .. فسوف ينتهى البحث ، إما إذا كان العنصر المطلوب أكبر من (له قيمة أكبر للمفتاح) عنصر الوسط ... فيجب أن يقع فى الجزء الأعلى من الجدول (بالنسبة للجداول المخزنة فى ترتيب تصاعدى) ، .. وبالمثل إذا كان العنصر المطلوب أقل من عنصر الوسط ... فيجب أن يقع النصف الأقل من الجدول . فى أى حالة من الحالتين .. يحذف نصف الجدول ؛ بدلاً من أخذه فى الاعتبار .

يستمر البحث بفحص عنصر الوسط لنصف الجدول المتبقى وتطبق نفس العملية ، مع التخلص من نصف الجدول الحالى . أخيراً .. لا يوجد إلا عنصراً واحداً ، سواء كان هو العنصر المطلوب ، أو لم يكن العنصر المطلوب موجوداً فى الجدول .

يبرمج البحث الثنائى فى الكويل باستخدام عبارة ابحث الكل SEARCH ALL (شكل ١٠ - ٤) . يجب أن يكون identifier-1 er - 1 هو اسم الجدول (معرفاً بجزء OCCURS) المراد البحث فيه ، كما يجب ألا يكون له دليل أو فهرس فى جزء SEARCH - ALL نفسه . يعمل جزء AT END كما يعمل فى البحث المتتابع ، بتنفيذه إذا فشل البحث الثنائى فى الجدول المحدد فقط . وإذا تحقق شرط WHEN أثناء البحث .. فتتخذ العبارة (أو العبارات) المصاحبة له وينتهى البحث . لاحظ أن شرط WHEN يجب أن يكون بنفس الصورة الموجودة فى شكل (١٠ - ٤) .

SEARCH ALL identifier-1
[AT END imperative-statement(s)-1]
WHEN ascending/descending-key-item EQUAL { identifier-2
literal
arithmetic-expression }
{ imperative-statement(s)-2 }
NEXT SENTENCE

شكل (١٠ - ٤)

بعد تنفيذ عبارة (عبارات) AT END ، أو عبارة (عبارات) WHEN (أو بعد تحقق شرط ATEND دون أن يحدد شرط AT END فى العبارة) ، يستمر التنفيذ بالعبارة التى تلى النقطة .. التى تنهى عبارة SEARCH ALL مباشرة .

عندما يجرى تشغيل على جدول باستخدام عبارة SEARCH ALL . يجب أن يحتوى جزء OCCURS الخاص به على جزء ASCENDING KEY ، أو DESCENDING KEY (شكل ١٠ - ٥) . من المهم أن يفهم إن هذا الجزء لا يتسبب فى ترتيب الجدول ، ولكنه يحدد الحقل المرتب الجدول على أساسه .

OCCURS integer-1 [TO integer-2] TIMES
[DEPENDING ON data-name-1]
{ ASCENDING
DESCENDING } KEY IS data-name-2
[INDEXED BY index-name-1 [index-name-2] ...]

شكل (١٠ - ٥)

مثال ١٠ - ٢٦ :

إذا كان المطلوب استرجاع وصف العمل لـ (3) INPUT-JOB - CODE PIC X من الجدول التالي:

```
01 JOB-CODES-TABLE.
   05 NUMBER-JOB-CODES      PIC S9(5)  COMP SYNC.
   05 JOB-CODES-MEANINGS    OCCURS 1 TO 100 TIMES
                              DEPENDING ON NUMBER-JOB-CODES
                              ASCENDING KEY IS JOB-CODE
                              INDEXED BY JOB-CODE-INDEX.
      10 JOB-CODE           PIC X(3).
      10 JOB-DESCRIPTION    PIC X(35).
```

افترض أن الجدول مرتب فعلا ترتيباً تصاعدياً؛ طبقا لحقل JOB - CODE، ونستطيع أن نستخدم البحث الثنائي كما يلي:

```
SEARCH ALL JOB-CODES-MEANINGS
AT END
  MOVE "*** INVALID JOB CODE ***" TO OUTPUT-JOB-DESCRIPTION
WHEN JOB-CODE (JOB-CODE-INDEX) EQUAL INPUT-JOB-CODE
  MOVE JOB-DESCRIPTION (JOB-CODE-INDEX) TO OUTPUT-JOB-DESCRIPTION
```

بعض الملاحظات :

- (١) يجب أن يظهر اسم الجدول ، كما هو معرف بجزء OCCURS ، في SEARCH ALL بدون دليل أو فهرس .
- (٢) لا يسرى جزء VARYING مع SEARCH ALL : فتغير SEARCH ALL دائما من أول فهرس معرف للجدول الذي يجرى البحث فيه ، ولا توجد حاجة لوضع قيمة ابتدائية للفهرس عند استخدام SEARCH ALL .
- (٣) لكي تعمل SEARCH ALL بطريقة صحيحة ، يجب أن يكون الجدول مرتبا فعلا ، كما هو مذكور في جزء KEY . وإذا لم يكن الجدول مرتبا بطريقة صحيحة .. فلن تظهر رسالة تحذيرية أو رسالة خطأ ، وإنما تنتج عبارة SEARCH ALL نتائج خاطئة ببساطة ؛ خاصة إذا كان الجدول الحالي ثابت الطول ، مع عدم جزء DEPENDING ؛ فبعض العناصر البالغة إجمالي عددها 100 تكون غير نشطة (نفايا) ، وهذا يجعل الجدول غير مرتب ككل ، ويتوقع أن تكون نتيجة SEARCH ALL نفايا كذلك . انظر المسالتين ٧٦ و ٧٧ من هذا الفصل ؛ لمعرفة كيف يرتب البرنامج جدولا غير مرتب .
- (٤) ترتيب الأجزاء داخل OCCURS غير قابل للتغيير في كويل IBM OS/VS (أي إنها يجب أن تكتب بنفس الترتيب المبين في شكل ١٠ - ٥) .

مثال ١٠ - ٢٧ :

يراد إدخال البيانات الموجودة في (3) INPUT-JOB - CODE PIC X ، وفي INPUT-JOB - DESCRIPTION (35) PIC X كعنصر جديد في جدول مثال (١٠ - ٢٦) ؛ لحفظ الجدول في ترتيب تصاعدي طبقا للحقل JOB - CODE ، ويجب أن تبحث في الجدول عن أول عنصر يكون أكبر من (بالنسبة لقيمة المفتاح key) العنصر الجديد ، ثم يوضع العنصر الجديد أمامه . إذا لم يوجد أى عنصر أكبر من العنصر الجديد .. فيجب وضع العنصر الجديد في نهاية الجدول . وإذا تساوى عنصر مع العنصر الجديد الذى وجد في الجدول .. فيتحقق شرط خطأ ، ويجب التعامل معه .

نرغب في اختبار JOB - CODE GREATER THAN WHEN من عبارة SEARCH ALL ، مقدم فقط لـ
JOB - CODE - EQUAL . وعلى هذا .. نجد أنفسنا مجبرين على استخدام SEARCH بدلا من SEARCH ALL ؛ بالرغم
من أن الجدول مرتب :

```

SET JOB-CODE-INDEX TO 1
SEARCH JOB-CODES-MEANINGS
  AT END
    ADD 1 TO NUMBER-JOB-CODES
    SET JOB-CODE-INDEX TO NUMBER-JOB-CODES
    MOVE INPUT-JOB-CODE TO JOB-CODE (JOB-CODE-INDEX)
    MOVE INPUT-JOB-DESCRIPTION TO
      JOB-DESCRIPTION (JOB-CODE-INDEX)
  WHEN
    JOB-CODE (JOB-CODE-INDEX) EQUAL INPUT-JOB-CODE
    PERFORM DUPLICATE-CODE-ERROR,
  WHEN

    JOB-CODE (JOB-CODE-INDEX) GREATER THAN INPUT-JOB-CODE
    ADD 1 TO NUMBER-JOB-CODES
    PERFORM MOVE-ENTRIES-UP
      VARYING MOVE-INDEX
        FROM NUMBER-JOB-CODES BY -1
        UNTIL MOVE-INDEX EQUAL JOB-CODE-INDEX
    MOVE INPUT-JOB-CODE TO JOB-CODE (JOB-CODE-INDEX)
    MOVE INPUT-JOB-DESCRIPTION TO
      JOB-DESCRIPTION (JOB-CODE-INDEX)
  .....
MOVE-ENTRIES-UP.
  MOVE JOB-CODES-MEANINGS (MOVE-INDEX - 1) TO
    JOB-CODES-MEANINGS (MOVE-INDEX)

```

تعليقات

(١) تغيير جزء INDEXED في شكل (١٠ - ٣٦) إلى الشكل التالي :

```

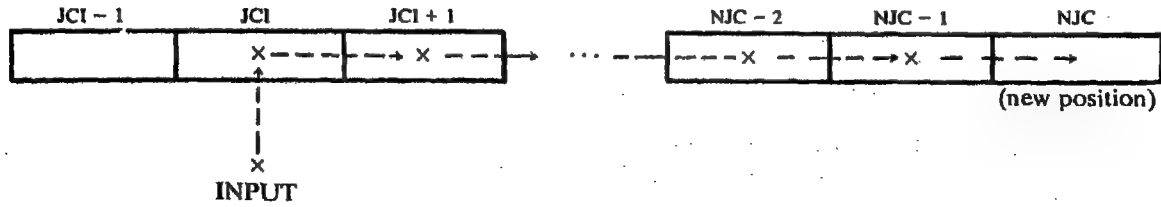
INDEXED BY JOB-CODE-INDEX
MOVE-INDEX.

```

(٢) إذا نتج عن البحث تحقق شرط AT END .. فإننا نعرف أن العنصر الجديد أكبر من كل العناصر الموجودة . وعلى هذا ..
فإننا نضيفه إلى نهاية الملف عن طريق (١) زيادة عنصر DEPENDING (ب) نقل المعلومات الجديدة إلى آخر عنصر
جديد ، نتج من الخطوة (١) .

(٣) إذا حدد البحث عنصرا مساويا للعنصر الجديد ، فإننا ننفذ PERFORM مقطعا للخطأ .

(٤) إذا حدد البحث قيمة JOB - CODE - INDEX تعدى لها HOB - CODE القيمة INPUT - JOB - CODE .. فإننا ننتج - عند ذلك - موقعا جديدا أخيرا في الجدول بزيادة NUMBER - JOB - CODES (انظر شكل ١٠ - ٦) . آخر عنصر في الجدول ، عند 1 - NUMBER - JOB - CODES الآن ، ينقل إلى الموقع الجديد ، ويأخذ - NUMBER - JOB - CODES - 2 مكانه ، وينقل العنصر JOB - CODE - INDEX إلى JOB - CODE - INDEX + 1 ، وتنتهي عبارة PERFORM .. تنقل بيانات المدخلات إلى JOB - CODE - INDEX . لاحظ أن الترتيب العكسي للنقل ضروري ، وإذا حاولنا البدء بنقل البيانات من JOB - CODE INDEX للأمام ، فتكون نتيجة ذلك أن تمحو البيانات كل العناصر التالية في الجدول .



شكل (١٠ - ٦)

أسئلة مراجعة :

- ١٠ - ١ اذكر عديداً من أمثلة جداول الأعمال .
- ١٠ - ٢ ناقش العوامل التي تأخذ في الاعتبار تحديد أقصى حجم لجدول .
- ١٠ - ٣ وضح استخدام الجداول في تشفير معلومات على هيئة رموز coded .
- ١٠ - ٤ ماذا يعني «توسيع الرمز» (الشفرة) خلال فحص الجدول؟
- ١٠ - ٥ ما الدليل وكيف يجب تعريفه ؟
- ١٠ - ٦ أين يمكن تعريف الجداول في جزء البيانات ؟
- ١٠ - ٧ ناقش استخدام جزء VALUE في وضع قيم ابتدائية في جداول مخزن العمل .
- ١٠ - ٨ تحت أي شروط .. يعمل MOVE ZERO TO ENTIRE - TABLE بطريقة صحيحة ؟
- ١٠ - ٩ متى يجب استخدام عبارة MOVE في مقطع ، ينفذ باستخدام PERFORM ؛ لوضع قيم ابتدائية لعناصر الجدول؟
- ١٠ - ١٠ ناقش أهمية حفظ عداد بعدد العناصر النشطة لمعظم الجداول . متى لا تكون هناك حاجة إلى مثل هذا العداد ؟
- ١٠ - ١١ لماذا يفضل عادة وضع قيم أولية لجدول من ملف ، بدلاً من استخدام VALUE , REDEFINES ؟
- ١٠ - ١٢ كيف تعرف الجداول ذات البعدين ؟
- ١٠ - ١٣ وضح قواعد استخدام دلائل أو فهارس لمحتويات جداول ذات بعدين .
- ١٠ - ١٤ ناقش استخدام AFTER ... VARYING ... PERFORM مع جداول ذات بعدين .

- ١٠ - ١٥ وضع كيف تعمل بيانات المدخلات كدلائل . متى يمكن أن يكون هذا مفيداً ؟
 - ١٠ - ١٦ ما أهمية التأكد من صحة حقول المدخلات المستخدمة كدلائل ؟
 - ١٠ - ١٧ ناقش كيفية ترجمة حقول المدخلات الحرفية عددية الى قيم دلائل ؟
 - ١٠ - ١٨ وضع معنى جزء OCCURS متداخل . ما القيد على OCCURS المتداخلة في الكويل ؟
 - ١٠ - ١٩ لماذا لا يحافظ استخدام الجداول متغيرة الطول على الذاكرة الرئيسية للكمبيوتر ؟ هل تستطيع الجداول متغيرة الطول أن تحفظ مكاناً في أى مكان آخر داخل نظام الكمبيوتر ؟
 - ١٠ - ٢٠ ما العلاقة بين السجلات متغيرة الطول والجداول متغيرة الطول ؟
 - ١٠ - ٢١ ناقش وضع عنصر DEPENDING داخل السجل المنطقي . هل يمكن أن يتبع الجدول متغير الطول ؟
 - ١٠ - ٢٢ اذكر القيود على جزء OCCURS ، عندما تكون الجداول متغيرة الطول مشمولة .
 - ١٠ - ٢٣ لماذا يجب تعريف الجدول متغير الطول في مخزن العمل ؟ هل يوفر هذا من مواقع ذاكرة الكمبيوتر ؟
 - ١٠ - ٢٤ ناقش الأخطاء التي يمكن الوقوع فيها عند نقل جداول متغيرة الطول .
 - ١٠ - ٢٥ لماذا لا يمكن استخدام WRITE ... FROM, ... READ ... INTO مع سجلات تحتوي على جداول متغيرة الطول ؟
 - ١٠ - ٢٦ ناقش طريقتين لحذف أحد العناصر من جدول .
 - ١٠ - ٢٧ ناقش دور رمز الحذف deletion code عند حذف محتويات منطقياً من جدول . ما آثار الحذف المنطقي على البرامج الأخرى التي تجري تشغيلاً على الجدول ؟
 - ١٠ - ٢٨ ناقش الخوارزمي الخاص بضغط جدول عند استخدام الحذف الواقعي (الطبيعي) .
 - ١٠ - ٢٩ ما الفهرس ؟ متى يجب استخدام الفهارس ؟ كيف تعرف الفهارس ؟
 - ١٠ - ٣٠ قارن قيم الفهرس مع قيم الدليل .
 - ١٠ - ٣١ ما حساب العنوان address calculation ؟
 - ١٠ - ٣٢ ما الجداول التي يجب استخدام الفهارس معها ؟
 - ١٠ - ٣٣ اذكر ثلاث طرق يمكن فيها تعديل الفهارس ، أو وضع قيم ابتدائية لها
 - ١٠ - ٣٤ ناقش استخدام عبارة SET . ما تكوينها ؟
 - ١٠ - ٣٥ ناقش كيف يمكن استخدام الفهارس في عبارة PERFORM ، و عبارة IF .
 - ١٠ - ٣٦ ما الفهرسة النسبية ؟ ناقش استخدامها .
 - ١٠ - ٣٧ هل هناك مكافئ للفهرسة النسبية في الدلائل ؟
 - ١٠ - ٣٨ وضع خوارزمي البحث المتتابع .
 - ١٠ - ٣٩ ناقش عبارة البحث . ما تكوينها ؟ وضع .
- VARYING ... (١)
- AT END ... (ب)

(ج) ... WHEN

(د) وضع قيمة ابتدائية لفهرس SEARCH .

- ١٠ - ٤٠ ناقش أهمية ترتيب كتابة أجزاء WHEN في عبارة SEARCH .
- ١٠ - ٤١ ما الجداول المتوازية parallel tables ؟ كيف ترتبط بجزء VARYING من عبارة SEARCH ؟
- ١٠ - ٤٢ ناقش كيف يمكن تطبيق SEARCH ، أو SEARCH ALL على جداول ذات بعدين .
- ١٠ - ٤٣ وضع خوارزمي البحث الثنائي . ما الشرط السابق الذي يجب تحقيقه لكي يعمل البحث الثنائي بطريقة صحيحة ؟
- ١٠ - ٤٤ ناقش الكفاءة النسبية للبحث المتتابع والبحث الثنائي . متى يجب استخدام كل منها ؟
- ١٠ - ٤٥ ناقش عبارة SEARCH ALL . كيف تختلف عن عبارة SEARCH ؟
- ١٠ - ٤٦ وضع استخدام ASCENDING KEY , DESCENDING KEY .
- ١٠ - ٤٧ يجب ألا تستخدم SEARCH ALL مع جدول متغير الطول ، إلا إذا كان معلوماً لماذا ؟
- ١٠ - ٤٨ ناقش خوارزمي لإدخال عنصر في جدول مرتب .
- ١٠ - ٤٩ ناقش خوارزمي لحذف عنصر من جدول حذفاً واقعياً .

مسائل محلولة

١٠ - ٥٠ ما الأخطاء الموجودة في تعريفات الجداول التالية :

- | | |
|---|---|
| (a) 01 DECK-OF-CARDS | PIC X(2)
OCCURS 52 TIMES. |
| (b) 01 DECK-OF-CARDS.
05 POKER-HAND

10 CARDS
15 RANK
15 SUIT
05 NUMBER-OF-PLAYERS | OCCURS 1 TO 10 TIMES
DEPENDING ON NUMBER-OF-PLAYERS.
OCCURS 5 TIMES.
PIC XX.
PIC X.
PIC S9(5) COMP SYNC. |
| (c) 01 POKER-HAND.
05 CARD

10 RANK
10 SUIT | OCCURS 5 TIMES
INDEXED BY CARD-INDEX
ASCENDING KEY IS RANK.
PIC XX.
PIC X. |
| (d) 01 POKER-TABLE.
05 NUMBER-OF-PLAYERS
05 POKER-HAND

10 NUMBER-DEALT
10 CARDS

15 RANK
15 SUIT | PIC S9(4) COMP SYNC.
OCCURS 1 TO 10 TIMES
DEPENDING ON NUMBER-OF-PLAYERS
INDEXED BY HAND-INDEX.
PIC S9(4) COMP SYNC.
OCCURS 1 TO 7 TIMES
DEPENDING ON NUMBER-DEALT
INDEXED BY CARD-INDEX.
PIC XX.
PIC X. |

- (a) يجب ألا يستخدم جزء OCCURS على المستوى 01 .
 (b) هناك عنصر بيانات NUMBER - OF - PLAYERS يتبع جدول متغير الطول في وصف السجل .
 (c) INDEXED BY يجب أن يتبع ASCENDING KEY (طبقاً لكويل IBM OS / VS) .
 (d) يتبع جزء OCCURS ... DEPENDING جزء OCCURS آخر . ومن الصحيح استخدام ما يلي :

10 CARDS OCCURS 7 TIMES
 INDEXED BY CARD-INDEX.

أى إنه يمكن لجدول ثابت الطول أن يتبع جدول متغير الطول (أو ثابت الطول كذلك) .

١٠ - ٥١ علق على الجدول التالي :

01 BASEBALL-TEAM
 05 STARTING-PLAYER OCCURS 9 TIMES
 INDEXED BY PLAYER-INDEX.
 10 PLAYER-NAME PIC X(25).
 10 PLAYER-POSITION PIC X(5).
 10 BATTING-AVERAGE PIC SV999 COMP-3.

يحتوى هذا الجدول على 9 عناصر نشطة دائماً . عادة ما يتغير عدد العناصر النشطة في جدول ، سواء كان هذا أثناء تنفيذ برنامج يجرى تشغيلاً على الجدول ، أو عند نقطة معينة خلال فترة استخدام البرنامج .
 ١٠ - ٥٢ عرف جدولاً ليحتوى على عدد أسطر الكويل التى تنتج في الاسبوع بواسطة كل مبرمج في إحدى المؤسسات . يعمل حالياً 55 مبرمجاً . وتخطط المؤسسة لتعين اثني عشر مبرمجاً في العام ، ثم تعيين 10 % أكثر في السنة التى بعدها . تقدر الفترة المتوقعة لاستخدام البرنامج الذى يجرى تشغيلاً على الجدول بأربع سنوات .

01 PROGRAMMER-PRODUCTIVITY.
 05 NUMBER-OF-PROGRAMMERS PIC S9(5) COMP SYNC.
 05 PROGRAMMER-DATA OCCURS 1 TO 94 TIMES
 DEPENDING ON
 NUMBER-OF-PROGRAMMERS
 INDEXED BY PROGRAMMER-INDEX.
 10 PROGRAMMER-ID PIC X(2).
 10 LINES-OF-COBOL PIC S9(3) COMP-3.

أول سنة يستخدم فيها البرنامج .. سيكون لدى المؤسسة 70 (15+55) مبرمجاً ، وفي السنة الثانية سيكون لديها 77 (70+7) مبرمجاً ، وفي السنة الثالثة يكون لديها 85 (77+8) مبرمجاً ، وفي السنة الرابعة يكون لديها 94 (85+9) مبرمجاً ، وعلى هذا .. يكون الجدول متغير الطول بحد أقصى 94 عنصراً .

١٠ - ٥٣ باستخدام جدول المسألة ٥١ .. أكتب شفرة لعرض DISPLAY الموقع ، الذي له أقل متوسط - AVER - BATTING

: AGE

INDEXED BY PLAYER - INDEX LOWEST - IN- حيث يفيد الفهرس الثاني . عدل جزء INDEXED ليصبح

: DEX عندئذ

```
SET LOWEST-INDEX TO 1
PERFORM FIND-LOWEST-AVERAGE
  VARYING PLAYER-INDEX FROM 2 BY 1
  UNTIL PLAYER-INDEX GREATER THAN 9
DISPLAY PLAYER-POSITION (LOWEST-INDEX)
```

.....
FIND-LOWEST-AVERAGE.

```
IF BATTING-AVERAGE (PLAYER-INDEX) LESS THAN
  BATTING-AVERAGE (LOWEST-INDEX)
  SET LOWEST-INDEX TO PLAYER-INDEX
```

١٠ - ٥٤ إذا كان ثمة قيد في المسألة السابقة على أقل متوسط .. هل يعرض DISPLAY البرنامج أول واحد يجده ، أم يعرض آخر واحد يجده ؟

حيث إن المؤثر العلاقي LESS THAN يستخدم .. فسوف يعرض DISPLAY أول متوسط منخفض .

١٠ - ٥٥ عدل المسألة رقم ٥٣ لعرض DISPLAY آخر متوسط ، يوجد كقيد على أقل متوسط .

استخدم NOT GREATER THAN بدلاً من LESS THAN في FIND - LOWEST - AVERAGE .

١٠ - ٥٦ حلل شفرة (3) PIC X - CODE LOCATION - INPUT إلى (30) PIC X - LOCATION - OUTPUT ،

مستخدماً المعلومات الموجودة في الجدولين المتوازيين التاليين .

01 LOCATION-CODE-TABLE.

05 LOCATION-CODE

```
PIC X(3)
OCCURS 80 TIMES
ASCENDING KEY
  LOCATION-CODE
INDEXED BY CODE-INDEX.
```

01 LOCATION-DESCRIPTION-TABLE.

05 LOCATION-DESCRIPTION

```
PIC X(30)
OCCURS 80 TIMES
INDEXED BY
  DESCRIPTION-INDEX.
```

حيث إن محتويات الجدولين متناظرة ، فإننا نستخدم (بافتراض أن LOCATION - CODE - TABLE مرتب فعلاً طبقاً لـ

LOCATION - CODE) ما يلي :

SEARCH ALL LOCATION-CODE

AT END

MOVE *** UNKNOWN LOCATION *** TO OUTPUT-LOCATION

WHEN

LOCATION-CODE (CODE-INDEX) EQUAL INPUT-LOCATION-CODE

SET DESCRIPTION-INDEX TO CODE-INDEX

MOVE LOCATION-DESCRIPTION (DESCRIPTION-INDEX)

TO OUTPUT-LOCATION

١٠ - ٥٧ أعد حل المسألة السابقة إذا لم يكن الجدول مرتباً .

يجب استخدام البحث المتتابع الآن :

```
SET CODE-INDEX
  DESCRIPTION-INDEX TO 1
SEARCH LOCATION-CODE
  VARYING DESCRIPTION-INDEX
  AT END
    MOVE *** UNKNOWN LOCATION *** TO OUTPUT-LOCATION
  WHEN
    LOCATION-CODE (CODE-INDEX) EQUAL INPUT-LOCATION-CODE
    MOVE LOCATION-DESCRIPTION (DESCRIPTION-INDEX)
      TO OUTPUT-LOCATION
```

نظراً لأن عنصر VARYING يخص جدولاً آخر .. فإنه يزداد بزيادة فهرس SEARCH (أول فهرس لجدول SEARCH) وعلى هذا يشير DESCRIPTION - INDEX و CODE - INDEX دائماً إلى المحتويات المناظرة .

١٠ - ٥٨ ماذا يحدث في الكوبل إذا انخفضت قيمة الدليل أو الفهرس لأقل من 1 (أول عنصر في الجدول) ، أو تعدت الحجم الحالي للجدول ؟

يمكن أن تثار مناطق الذاكرة التي تسبق أو تتبّع الجدول . إذا كانت هناك أدنى فرصة لأن تصبح قيمة الدليل أو الفهرس غير صحيحة .. فيجب أن يختبرها البرنامج قبل استخدامها في الاتصال بالجدول .

١٠ - ٥٩ انقد ما يلي :

```
01 PROJECT-ASSIGNMENTS.
  05 PROJECT-VALUES.
    10 FILLER    PIC X(11)    VALUE "MARYX250100".
    10 FILLER    PIC X(11)    VALUE "MIKE2A70553".
    10 FILLER    PIC X(11)    VALUE "JOHNRS20050".
  05 PROGRAMMER-INFO
    REDEFINES PROJECT-VALUES
    OCCURS 3 TIMES
    INDEXED BY PROGRAMMER-INDEX.
    10 FIRST-NAME    PIC X(4).
    10 PROJECT-CODE   PIC X(3).
    10 HOURS-WORKED   PIC S9(3)V9    COMP-3.
```

١ - لا يتفق جزء VALUE مع وصف محتوى الجدول ؛ إذ يفترض أن تكون HOURS - WORKED حقلاً من نوع COMP - 3 ، ولكن جزء VALUE حدد لها الاستخدام DISPLAY .

٢ - حيث يمكن أن تتغير المعلومات الموجودة في الجدول بصفة متكررة ، فيكون استخدام REDEFINES VALUE في وضع القيم الابتدائية .. اختباراً ضعيفاً جداً . ومن الأفضل كثيراً حفظ معلومات الجدول في ملف قرص ؛ حيث يمكن تغييره ببساطة ، ويمكن نقل قيم الجدول منه كمدخلات كلما دعت الحاجة لذلك .

٣ - حيث إن عدد المبرمجين يتغير ، فيجب أن يوجد عداد مناصب للجدول ؛ ليحدد عدد العناصر النشطة . ومثل هذا العداد يعرف عادة كعنصر على المستوى 05 تحت PROJECT - ASSIGNMENTS .

١٠ - ٦٠ صحح PROJECT - VALUES في المسألة السابقة لمواجهة الاعتراض رقم (1) في حل المسألة السابقة .

05 PROJECT-VALUES.

```
10 FILLER PIC X(7) VALUE "MARYX25".
10 FILLER PIC S9(3)V9 COMP-3 VALUE +10.0.
10 FILLER PIC X(7) VALUE "MIKE2A7".
10 FILLER PIC S9(3)V9 COMP-3 VALUE +55.3.
10 FILLER PIC X(7) VALUE "JOHNRS2".
10 FILLER PIC S9(3)V9 COMP-3 VALUE +5.0.
```

١٠ - ٦١ (1) هل ما يلي يعمل بطريقة صحيحة مع جداول المسألة ١٠ - ٥٦ ؟

MOVE SPACES TO LOCATION-CODE-TABLE
LOCATION-DESCRIPTION-TABLE

(ب) هل ما يلي يعمل بطريقة صحيحة مع جداول المسألة ١٠ - ٥٢ ؟

MOVE SPACES TO PROGRAMMER-DATA

(1) نعم ، حيث إن لكل المحتويات الوصف : PIC X (m) DISPLAY .

(ب) لا . أولا .. تتطلب PROGRAMMER - DATA دليلا أو فهرسا عند استخدامها في عبارة MOVE . ثانيا .. الحقول

المجدولة هي (2) PIC X ، (3) COMP - 3، PIC S9 (3) ، بحيث إنه لا يمكن لعبارة MOVE أن تضع قيما ابتدائية لهما .

١٠ - ٦٢ اكتب شفرة لوضع فراغات واصفان على التوالي لـ PROGRAMMER - ID ، COBOL - OF - LINES في

المسألة ١٠ - ٥٢ .

```
PERFORM CLEAR-PROGRAMMER-DATA
  VARYING PROGRAMMER-INDEX FROM 1 BY 1
  UNTIL PROGRAMMER-INDEX GREATER THAN
    NUMBER-OF-PROGRAMMERS
```

.....

CLEAR-PROGRAMMER-DATA.

```
MOVE SPACES TO PROGRAMMER-ID (PROGRAMMER-INDEX)
MOVE ZEROS TO LINES-OF-COBOL (PROGRAMMER-INDEX)
```

١٠ - ٦٣ يعرف جدولاً يمكن أن يحتوى حتى 100 عنصر من (5) PIC X ID - PART و (4) COMP S9 PIC QUANTITY

لكل من : WAREHOUSE - LOCATION ، الذى له PIC X ويمكن أن يصل عدده إلى 20 .

01 WAREHOUSE-CONTENTS.

```
05 NUMBER-LOCATIONS
05 LOCATION-DATA
```

```
PIC S9(5). COMP SYNC.
OCCURS 1 TO 20 TIMES
DEPENDING ON NUMBER-LOCATIONS
INDEXED BY LOCATION-INDEX.
```


10	WAREHOUSE-LOCATION	PIC X(3).
10	NUMBER-PARTS	PIC S9(5) COMP.
10	PARTS-DATA	OCCURS 100 TIMES
		INDEXED BY PART-INDEX.
15	PART-ID	PIC X(5).
15	QUANTITY	PIC S9(4) COMP.

OCCURS 1 TO 100 TIMES DEPENDING ON PARTS - DATA على أنها
 NUMBER - PARTS فلا يمكن عمل جدول متغير الطول ، كجزء من أى جدول آخر.

١٠ - ٦٤ أى أسماء بيانات من المسألة السابقة يمكن أن يظهر فى

(أ) عبارة SEARCH ALL ؟ (ب) فى عبارة SEARCH ؟

(أ) لا شىء ، حيث إنها ليست أجزاء مفتاح تصاعدى أو تنازلى .

(ب) LOCATION - DATA أو PARTS - DATA (لاحظ أن الجدول يجب أن يكون مفهرسا لاستخدامه فى عبارات ،
 SEARCH ، أو SEARCH ALL

١٠ - ٦٥ اكتب كل اسم بيانات من المسألة ٦٣ ، كما يجب أن يظهر فى عبارة MOVE .

WAREHOUSE-CONTENTS; NUMBER-LOCATIONS; LOCATION-DATA (LOCATION-
 INDEX); WAREHOUSE-LOCATION (LOCATION-INDEX); NUMBER-PARTS (LOCATION-
 INDEX); PARTS-DATA (LOCATION-INDEX PART-INDEX); PART-ID (LOCATION-INDEX
 PART-INDEX); QUANTITY (LOCATION-INDEX PART-INDEX).

١٠ - ٦٦ بين كيف يضاف 10% إلى كل عنصر QUANTITY من جدول المسألة ٦٣ .

PERFORM ADD-TEN-PERCENT
 VARYING LOCATION-INDEX FROM 1 BY 1
 UNTIL LOCATION-INDEX GREATER THAN NUMBER-LOCATIONS
 AFTER PART-INDEX FROM 1 BY 1
 UNTIL PART-INDEX GREATER THAN
 NUMBER-PARTS (LOCATION-INDEX)

ADD-TEN-PERCENT.

COMPUTE QUANTITY (LOCATION-INDEX PART-INDEX) =
 QUANTITY (LOCATION-INDEX PART-INDEX) * 1.10

لاحظ أنه يجب فهرسة NUMBER - PARTS كما هو مبين : فهي تصبح عنصراً في الجدول .

١٠ - ٦٧ أكتب شفرة تعرض DISPLAY كل مواقع المخازن التي تخزن العنصر المعروف بواسطة INPUT - PART - ID

(5) IPC X استخدم جدول المسألة ٦٣ .

```
PERFORM FIND-WHERE-PART-IS
  VARYING LOCATION-INDEX FROM 1 BY 1
  UNTIL LOCATION-INDEX GREATER THAN NUMBER-LOCATIONS
```

```
.....
FIND-WHERE-PART-IS.
  SET PART-INDEX TO 1
  SEARCH PARTS-DATA
  AT END
    NEXT SENTENCE
  WHEN
    INPUT-PART-ID EQUAL
      PART-ID (LOCATION-INDEX PART-INDEX)
      DISPLAY WAREHOUSE-LOCATION (LOCATION-INDEX)
```

١٠ - ٦٨ إذا كان معروفاً السجلات المنطقية التالية :

```
FD SHIPPING-FILE ...
01 SHIPPING-RECORD.
   05 CUSTOMER-ID      PIC X(6).
   05 PRODUCT-ID       PIC X(4).
   05 SHIPPING-CLASS   PIC S9(2).
   05 WEIGHT-CLASS     PIC S9(2).
   05 SHIP-TO-ADDRESS  PIC X(50).
```

أفرض ان جدول تكلفة الشحن تم تعريفه وتحمله في مخزن العمل :

```
01 SHIPPING-CHARGE-TABLE.
   05 NUMBER-SHIPPING-CLASSES PIC S9(5) COMP SYNC.
   05 SHIPPING-CLASS-ENTRY    OCCURS 30 TIMES
                                INDEXED BY SHIPPING-INDEX.
                                10 NUMBER-WEIGHT-CLASSES PIC S9(5) COMP.
                                10 WEIGHT-CLASS-ENTRY    OCCURS 15 TIMES
                                                                INDEXED BY WEIGHT-INDEX.
                                15 SHIPPING-CHARGE        PIC S9(3)V99 COMP-3.
```

بافتراض أن SHIPPING - CLASS معرف بأنه عددي صحيح يقع بين 1 و 30 ، وأن WEIGHT - CLASS معرف

بأنه عددي صحيح يقع بين 1 و 15 . أكتب شفرة تضع تكلفة الشحن المناسبة في CUSTOMER - SHIP - CHARGE ، إذا

لم يمكن تحديد موقع للتكلفة ، ضع صفراً في CUSTOMER - SHIP - CHARGE .

```

IF SHIPPING-CLASS LESS THAN 1 OR GREATER THAN
  NUMBER-SHIPPING-CLASSES
  MOVE ZERO TO CUSTOMER-SHIP-CHARGE
ELSE IF WEIGHT-CLASS LESS THAN 1 OR GREATER THAN
  NUMBER-WEIGHT-CLASSES (SHIPPING-CLASS)
  MOVE ZERO TO CUSTOMER-SHIP-CHARGE
ELSE
  MOVE SHIPPING-CHARGE (SHIPPING-CLASS WEIGHT-CLASS)
  TO CUSTOMER-SHIP-CHARGE

```

١٠ - ٦٩ راجع المسألة ٦٨ لاستخدام فهرس بدلا من دلائل .

```

IF SHIPPING-CLASS LESS THAN 1 OR GREATER THAN
  NUMBER-SHIPPING-CLASSES
  MOVE ZERO TO CUSTOMER-SHIP-CHARGE
ELSE
  SET SHIPPING-INDEX TO SHIPPING-CLASS
  IF WEIGHT-CLASS LESS THAN 1 OR GREATER THAN
    NUMBER-WEIGHT-CLASSES (SHIPPING-INDEX)
    MOVE ZERO TO CUSTOMER-SHIP-CHARGE
  ELSE
    SET WEIGHT-INDEX TO WEIGHT-CLASS
    MOVE SHIPPING-CHARGE (SHIPPING-INDEX WEIGHT-INDEX)
    TO CUSTOMER-SHIP-CHARGE

```

١٠ - ٧٠ حدد الخطأ فيما يلي ، وحدد كيف يصحح .

```

SET A-TABLE-INDEX TO 10
SEARCH A-TABLE
  AT END NEXT SENTENCE

WHEN A-TABLE-ENTRY (A-TABLE-INDEX) EQUAL SPACES
  SEARCH ALL ANOTHER-TABLE
  WHEN ANOTHER-KEY (ANOTHER-INDEX) EQUAL ZERO
  PERFORM WHOOPEE

```

من المسموح به بدء البحث المتتابع عند أى نقطة فى الجدول . وعلى هذا بافتراض أن A - TABLE به عشرة عناصر على الأقل .. فإن عبارة SET تكون صحيحة . ولكن يجب أن يحتوى جزء WHEN على عبارة أمرية فقط ، وتصنف SEARCH ALL بأنها عبارة شرطية . ويمكن أن تكون الشفرة المناسبة كما يلي :

```
SET A-TABLE-INDEX TO 10
SEARCH A-TABLE
  AT END NEXT SENTENCE
  WHEN A-TABLE-ENTRY (A-TABLE-INDEX) EQUAL SPACES
  PERFORM SEARCH-ANOTHER
.....
```

```
SEARCH-ANOTHER.
  SEARCH ALL ANOTHER-TABLE
  WHEN ANOTHER-KEY (ANOTHER-INDEX) EQUAL ZERO
  PERFORM WHOOPEE
```

١٠ - ٧١ تغيرت المسألة ٦٨ بحيث SHIPPING - CLASS أصبح له (4) PIC X كما أن WEIGHT - CLASS أصبح له (3) PIC X ، ويشمل SHIPPING - CHARGE - TABLE عنصراً على المستوى 01 ، وهو SHIPPING - CLASS - ID ، وله (4) PIC X وعنصراً على المستوى 15 ، وهو WEIGHT - CLASS - ID وله (3) PIC X . أعد كتابة الشفرة لتحويل المدخلات الحرفية عديدة إلى فهارس مناسبة .

```
SET SHIPPING-INDEX TO 1
SEARCH SHIPPING-CLASS-ENTRY
  AT END
  MOVE ZERO TO CUSTOMER-SHIP-CHARGE
  WHEN
    SHIPPING-CLASS-ID (SHIPPING-INDEX) EQUAL
    SHIPPING-CLASS
    PERFORM LOCATE-WEIGHT-CLASS
.....
LOCATE-WEIGHT-CLASS.
  SET WEIGHT-INDEX TO 1
  SEARCH WEIGHT-CLASS-ENTRY
  AT END
  MOVE ZERO TO CUSTOMER-SHIP-CHARGE
  WHEN
    WEIGHT-CLASS-ID (SHIPPING-INDEX WEIGHT-INDEX)
    EQUAL WEIGHT-CLASS
    MOVE SHIPPING-CHARGE (SHIPPING-INDEX WEIGHT-INDEX)
    TO CUSTOMER-SHIP-CHARGE
```

١٠ - ٧٢ بمعرفة الجدول التالي (المسألة ٦٨) :

01 SHIPPING-TABLE.	
05 NUMBER-CLASSES	PIC S9(5) COMP SYNC.
05 CLASS-ENTRY	OCCURS 1 TO 80 TIMES
	DEPENDING ON NUMBER-CLASSES
	ASCENDING KEY CLASS-ID
	INDEXED BY CLASS-INDEX.
10 CLASS-ID	PIC X(4).
10 NUMBER-WEIGHTS	PIC S9(5) COMP.
10 WEIGHT-ENTRY	OCCURS 15 TIMES
	DESCENDING KEY WEIGHT-ID
	INDEXED BY WEIGHT-INDEX.
15 WEIGHT-ID	PIC X(3).
15 CHARGE	PIC S9(3)V99 COMP-3.

راجع شفرة المسألة السابقة ؛ بحيث تستخدم بحثاً ثنائياً كلما كان ذلك ممكناً مع إنهاء البحث التتابعى عندما ، يتعدى فهرس البحث عدد عناصر الجدول النشطة .

```

SEARCH ALL CLASS-ENTRY
  AT END
    MOVE ZERO TO CUSTOMER-SHIP-CHARGE
  WHEN
    CLASS-ID (CLASS-INDEX) EQUAL SHIPPING-CLASS
    PERFORM LOCATE-WEIGHT-CLASS
.....
LOCATE-WEIGHT-CLASS.
  SET WEIGHT-INDEX TO 1
  SEARCH WEIGHT-ENTRY
    AT END
      MOVE ZERO TO CUSTOMER-SHIP-CHARGE
    WHEN
      WEIGHT-INDEX GREATER THAN NUMBER-WEIGHTS (CLASS-INDEX)
      MOVE ZERO TO CUSTOMER-SHIP-CHARGE
    WHEN
      WEIGHT-ID (CLASS-INDEX WEIGHT-INDEX) EQUAL WEIGHT-CLASS
      MOVE CHARGE (CLASS-INDEX WEIGHT-INDEX)
        TO CUSTOMER-SHIP-CHARGE

```

لاحظ أن هذا الحل أكثر كفاءة عن الحل الموجود فى المسألة السابقة .

١٠ - ٧٣ لماذا لا تستخدم SEARCH ALL فى البحث فى الجدول WEIGHT - ENTRY فى المسألة السابقة ؟ ؛ لأنه جدول ذا طول ثابت لا يكون مملوء دائماً ، وعلى هذا لا يكون الجدول - ككل - مخزناً بالفعل .

١٠ - ٧٤ بمعرفة جداول المسألة رقم ٥٦ .. (أ) ماذا ينقل فعلاً بواسطة SET DESCRIPTION - INDEX TO 1 إلى ؟ DE-

SCRIPTION - INDEX (ب) ماذا يضاف بواسطة SET DESCRIPTION - INDEX UP BY 1 إلى DE ؟
SCRIPTION - INDEX

(أ) صفر : يمثل الفهرس الإزاحة داخل الجدول اللازمة .. للوصول إلى العنصر المطلوب . العنصر one صفر بايت ، أبعد من بداية الجدول .

(ب) 30 : تزداد أو تقل الفهارس زيادة فى معدل الخطوات بمضاعفات طول العنصر - طول عنصر واحد لكل موقع جدول .

١٠ - ٧٥ قم بمضاهاة حسابات عناوين لغة الآلة اللازمة لعمل الدلائل مع نظيرتها اللازمة لعمل الفهارس ، لجدول ذى بعد واحد .

عندما تكون K دليلاً ، فإنها تكون رقم موقع العنصر .

$$\text{address ENTRY (K)} = \text{address-start-of-table} + (K - 1) * \text{entry-length}$$

وعندما تكون K فهرساً .. فإنها تكون إزاحة من بداية الجدول :

$$\text{address ENTRY (K)} = \text{address-start-of-table} + K$$

١٠ - ٧٦ ماذا يحدث إذا أريد عمل بحث ثنائي لجداول مرتب من الأصل ؟ أحد الحلول .. إذا كانت قيم الجداول مدخلات من ملف ، .. فالحل عمل ترتيب إدخال insertion sort ، الذي يوضع فيه كل عنصر في موقعه المناسب ، في الجدول الكامل جزئيا ، كما لو كان العنصر مدخلا من الملف . طبق هذه الطريقة :

```
FD INPUT-FILE...
01 INPUT-RECORD.
   05 INPUT-CODE      PIC X(3).
   05 INPUT-DESCRIPTION PIC X(35).

01 JOB-CODES-TABLE.
   05 NUMBER-CODES     PIC S9(5) COMP SYNC.
   05 CODE-ENTRY       OCCURS 1 TO 100 TIMES
                       DEPENDING ON NUMBER-CODES
                       ASCENDING KEY IS JOB-CODE
                       INDEXED BY CODE-INDEX
                       MOVE-INDEX.
      10 JOB-CODE      PIC X(3).
      10 JOB-DESCRIPTION PIC X(35).

BUILD-TABLE.
  OPEN INPUT INPUT-FILE
  MOVE "NO" TO END-FILE-SW
  PERFORM GET-NEXT-RECORD
  IF END-FILE-SW EQUAL "NO"
    MOVE 1 TO NUMBER-CODES
    MOVE INPUT-CODE      TO JOB-CODE (1)
    MOVE INPUT-DESCRIPTION TO JOB-DESCRIPTION (1)
    PERFORM GET-NEXT-RECORD
    PERFORM INSERT-IN-SORTED-ORDER
    UNTIL END-FILE-SW EQUAL "YES" OR NUMBER-CODES
      EQUAL 100
  ELSE
    PERFORM FILE-EMPTY-ERROR

  IF END-FILE-SW NOT EQUAL "YES"
    PERFORM TABLE-TOO-SMALL-ERROR

  CLOSE INPUT-FILE
  .....
  INSERT-IN-SORTED-ORDER.
```

(جزء من هذا هو موجود في مثال ١٠ - ٢٧)

١٠ - ٧٧ هناك طريقة أخرى أقل كفاءة لعمل الترتيب ، تعرف بترتيب الفقاعة bubble sort ، يقارن فيها زوج من العناصر المتجاورة .. ويحدث تبادل بينهما إذا كانا غير مرتبين . من الواضح .. إنه إذا انتقلنا خلال الجدول لعمل المقارنة والتبادل بصورة كافية .. فيصبح الجدول مرتبا . اكتب شفرة لتنفيذ ترتيب الفقاعة على جدول المسألة السابقة ، الذي يجب أن يفترض أنه أعد فعلا .

نحن نعرف فهرسا آخر للجدول SORT - INDEX ، ونعرف كذلك عنصرى بيانات : HAD - TO - ENTRY - HOLD ، SWAP ، لهما الشكل (3) PIC X ، (38) PIC X على التوالى .

```
SET STOP-INDEX TO NUMBER-CODES
SET STOP-INDEX DOWN BY 1
MOVE "YES" TO HAD-TO-SWAP
PERFORM MAKE-A-PASS
    UNTIL STOP-INDEX NOT GREATER THAN ZERO
    OR HAD-TO-SWAP EQUAL "NO"
```

```
.....
MAKE-A-PASS.
    MOVE "NO" TO HAD-TO-SWAP
    PERFORM COMPARE-AND-SWAP
        VARYING CODE-INDEX FROM 1 BY 1
        UNTIL CODE-INDEX GREATER THAN STOP-INDEX
    SET STOP-INDEX DOWN BY 1
```

```
.....
COMPARE-AND-SWAP.
    IF JOB-CODE (CODE-INDEX) GREATER THAN
        JOB-CODE (CODE-INDEX+1)
        MOVE "YES" TO HAD-TO-SWAP
        MOVE CODE-ENTRY (CODE-INDEX) TO HOLD-ENTRY
        MOVE CODE-ENTRY (CODE-INDEX+1) TO
            CODE-ENTRY (CODE-INDEX)
        MOVE HOLD-ENTRY TO CODE-ENTRY (CODE-INDEX+1)
```

١٠ - ٧٨ اذكر طريقة أكثر كفاءة لتحميل الجدول فى مثال (١٠ - ٢١) .

حيث إن TABLE - VALUES - RECORD له نفس الشكل التخطيطى تماماً مثل عنصر WEIGHT - RANGE فى
TABLE - CHARGE - FREIGHT .. يمكن استخدام نقل MOVE مجموعة : لنقل محتويات العنصر فى عبارة واحدة .
وعلى هذا .. يمكن كتابة LOAD - TABLE كما يلى (مع إلغاء الحاجة إلى MOVE FREIGHT - CHARGES) :

```
LOAD-TABLE.
    MOVE TABLE-VALUES-RECORD TO
        WEIGHT-RANGE (NUMBER-WEIGHT-ENTRIES)
    ADD 1 TO NUMBER-WEIGHT-ENTRIES
    PERFORM GET-TABLE-RECORD
```


الفصل الحادى عشر

تشغيل الملفات تتابعيا

Sequential File Processing

اعتدنا الآن على : (١) إنتاج (٢) واسترجاع سجلات منطقية من ملف ، يكون تنظيمه تتابعيا - ORGANIZATION IS SEQUENTIAL ، وتعرف كذلك (٣) استخدام منطقة حالة الملف FILE STATUS فى اكتشاف الأخطاء فى تشغيل الملفات تتابعيا . فيما يلى توضيح أكثر لهذا ، كتوضيح أخير .

مثال ١١ - ١ :

فى نظم IBM يحدد رمز الحالة "00" إتماما ناجحا ، ويحدد "10" نهاية ملف ، ويحدد "30" خطأ دائما فى نظم مكونات المدخلات أو المخرجات ، ويحدد "92" خطأ منطقيا فى البرنامج . وعلى هذا .. فإذا سمي جزء FILE STATUS من عبارة SELECT عنصر من مخزن العمل STATUS - CODE - AREA PIC X يمكن أن يكون لدينا ما يلى فى جزء الإجراءات .

```
OPEN INPUT ANY-SEQUENTIAL-FILE
IF STATUS-CODE-AREA NOT EQUAL "00"
  DISPLAY "CANNOT OPEN FILE--ABORTING RUN"
  MOVE "YES" TO PROGRAM-ABORT-SW
```

```
.....
READ ANY-SEQUENTIAL-FILE
IF STATUS-CODE-AREA NOT EQUAL "00"
  IF STATUS-CODE-AREA EQUAL "10"
    MOVE "YES" TO END-FILE-SW
  ELSE IF STATUS-CODE-AREA EQUAL "30"
    DISPLAY "HARDWARE ERROR READING FILE"
    MOVE "YES" TO IGNORE-RECORD-SW
```

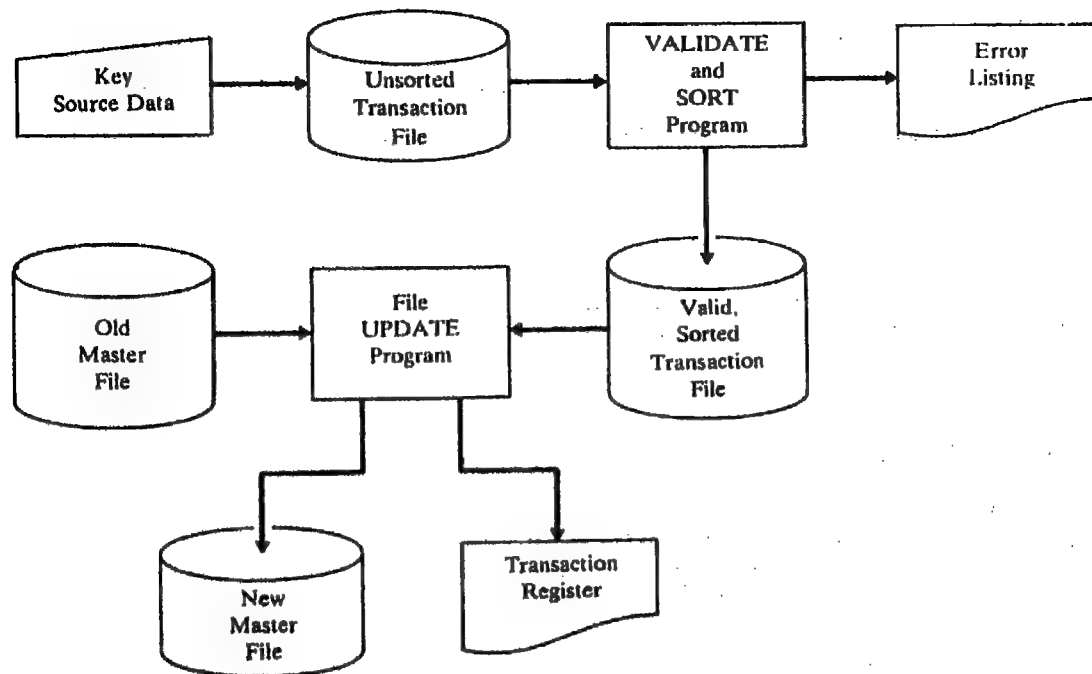
```
.....
CLOSE ANY-SEQUENTIAL-FILE
IF STATUS-CODE-AREA NOT EQUAL "00"
  DISPLAY "CLOSE FAILED--RUN DIAGNOSTIC ON FILE"
```

١١ - ١ تجديد الملفات التتابعية

تحفظ الملفات الرئيسية المرتبة تتابعيا دائما مرتبة طبقا لحقل رئيسى أو حقل مفتاح (القسم التاسع - الفصل العاشر) - وهذه حقيقة مهمة جدا عند تجديد updating الملف . وتدخل السجلات المنطقية المحتوية على معلومات مطلوب إضافتها أو حذفها من الملف الرئيسى عادة فى ملف تتابعى منفصل : ملف العمليات الجارية transaction . ويحدد كل سجل عمليات جارية المفتاح لسجل الملف الرئيسى الذى يضاف أو يجرى عليه تغيير أو يحدف ، ويرتب ملف العمليات الجارية نفسه طبقا لنفس المفتاح . ويوضح التجديد النمطى لملف تتابعى فى خريطة مسار فى شكل (١١ - ١) ، ويشتمل على الخطوات التالية :

١ - إما أن تجمع مستندات المصدر خلال فترة زمنية معينة ، ليتم إدخال الدفعة batch كلها فى ملف عمليات جارية تتابعى ، على قرص مرة واحدة ، وإما أن يتم إدخال العمليات الجارية عبر نهايات طرفية ، فى وسط الخط المفتوح كما تحدث ، إلا أن البرنامج الذى يعمل اتصالات مع النهايات الطرفية - بدلا من تشغيل هذه العمليات الجارية - يجمعها ببساطة على هيئة دفعة فى ملف عمليات جارية على قرص .

٢ - فى أى من الحالتين .. يكون لدينا ملف عمليات جارية تتابعى ، يكون قد اختبر ورتب فى نفس تتابع الملف الرئيسى المطلوب تجديده . كما يجب تصحيح العمليات الجارية الخاطئة ، والعودة إلى الخطوة الأولى ثانية .



شكل (١١ - ١)

٣ - تصبح العمليات الجارية التى أجرى لها تصحيح ، ورتبت كذلك مدخلات الآن لبرنامج تجديد ملف تتابعى ، يسترجع سجلات الملف الرئيسى الموجود ، ويجرى التغييرات المحددة ، وينتج ملفاً رئيسياً جديداً new master file مجدداً ، ويحفظ الملف الرئيسى القديم old master file فى هذه العملية .

٤ - يحفظ الملف الرئيسى القديم ، و ملف العمليات الجارية فى مكان آمن لأغراض الاحتياط backup ؛ فإذا حدث أى شئ للملف الرئيسى الجديد .. فإنه يمكن إعادة انتاجه بتنفيذ برنامج التجديد مرة أخرى . وتعرف هذه الطريقة بطريقة الجدود grandparent والاباء parent والاطفال child للاختياطى ؛ حيث يمكن حفظ أى عدد من أجيال الملف الرئيسى القديم .

٥ - ينتج برنامج تجديد الملف التتابعى المنفذ فى الخطوة ٣ عادة تقريراً مطبوعاً ، يسمى مسجل العمليات الجارية - transac-tion register ؛ يسرد نتائج تطبيق كل عملية جارية على الملف الرئيسى . لاحظ أنه بالرغم من أن العملية الجارية يمكن أن تكون صحيحة فى حد ذاتها ، إلا إنها يمكن أن تكون خطأ بالنسبة للملف الرئيسى (مثل : يمكن وجود عملية جارية ، محددة بطريقة صحيحة لسجل ملف رئيسى غير موجود) . ويجب تصحيح مثل هذه الأخطاء ، وإعادة الدورة مرة أخرى من الخطوة رقم (١) .

١١ - ٢ خوارزمى خط الاتزان

يخطط شكل (١١ - ٢) بشفرة شبيهة تشبة الكويل ، وتعد إحدى الطرق الأكثر كفاءة لتوفيق السجلات فى ملف العمليات الجارية . المرتب مع نظائرها فى الملف الرئيسى المرتب ، وهى تسمى بخوارزمى خطط الاتزان - balanced - line algo-rithm . قد يعالج الخوارزمى أكثر من سجل عمليات جارية واحد لكل سجل رئيسى ، وهذا موقف شائع عندما تكون العمليات الجارية على هيئة دفعات فى فترة زمنية . ويجب ملاحظة أن كل تفاصيل تشغيل حقول السجلات تترك لمقاطع (لأجزاء) أدنى مستوى . وعلى هذا .. يمكن استخدام أجزاء المستوى العلوى فى عديد من البرامج المختلفة لتطبيقات مختلفة .

يوجد قلب الخوارزمى فى مقطع UPDATE - MASTER - FILE ، ودائماً ما يكون LOW - KEY مساوياً أو أقل من TRANSACTION - KEY و OLD - MASTER - KEY ؛ فإذا كان مفتاح السجل OLD - MASTER - RECORD أقل من أو يساوى TRANSACTION - KEY (يتساوى LOW - KEY مع OLD - MASTER - KEY) ؛ فينقل عند ذلك - OLD MASTER - RECORD الى WS - NEW - MASTER - RECORD ، وهو منطقة مخزن عمل .. تبنى فيها سجلات الملف الرئيسى الجديد . يعد MASTER - READY - FOR - OUTPUT مفتاح برنامج ، يذكر ما إذا كانت هناك بيانات نشطة فى منطقة بناء هذا السجل أم لا .. يجب أن يتحقق القارئ (بإجراء اختبار مكتبى) أنه من المستحيل تنفيذ - MOVE OLD MASTER - RECORD TO WS - NEW - MASTER - RECORD عندما يكون هناك سجل نشط فى مرحلة البناء فى منطقة WS - NEW - MASTER - RECORD . وبمجرد نقل OLD - MASTER - RECORD إلى منطقة بناء السجل الرئيسى الجديد ، يقرأ السجل الرئيسى القديم التالى فى الذاكرة الاحتياطية لمدخلات OLD - MASTER - RECORD .

BALANCED-LINE.

```

MOVE "NO" TO MASTER-READY-FOR-OUTPUT
OPEN INPUT TRANSACTION-FILE
                OLD-MASTER-FILE
                OUTPUT NEW-MASTER-FILE
                ERROR-REGISTER
PERFORM GET-NEXT-TRANSACTION
PERFORM GET-NEXT-MASTER
PERFORM DETERMINE-LOW-KEY
PERFORM UPDATE-MASTER-FILE
                UNTIL LOW-KEY EQUAL HIGH-VALUES
CLOSE TRANSACTION-FILE
                OLD-MASTER-FILE
                NEW-MASTER-FILE
                ERROR-REGISTER
STOP RUN

```

UPDATE-MASTER-FILE.

```

IF LOW-KEY EQUAL OLD-MASTER-KEY
    MOVE OLD-MASTER-RECORD TO WS-NEW-MASTER-RECORD
    MOVE "YES" TO MASTER-READY-FOR-OUTPUT
    PERFORM GET-NEXT-MASTER

IF LOW-KEY EQUAL TRANSACTION-KEY
    PERFORM PROCESS-A-TRANSACTION
    UNTIL LOW-KEY NOT EQUAL TRANSACTION-KEY

IF MASTER-READY-FOR-OUTPUT EQUAL "YES"
    WRITE NEW-MASTER-RECORD FROM WS-NEW-MASTER-RECORD
    MOVE "NO" TO MASTER-READY-FOR-OUTPUT

PERFORM DETERMINE-LOW-KEY

```

PROCESS-A-TRANSACTION.

```

IF TRANSACTION-CODE-IS-ADD
    IF MASTER-READY-FOR-OUTPUT EQUAL "YES"
        PERFORM DUPLICATE-KEY-ERROR
    ELSE
        PERFORM MOVE-TRANS-INFO-TO-WS-NEW-MASTER
        MOVE "YES" TO MASTER-READY-FOR-OUTPUT
ELSE IF TRANSACTION-CODE-IS-CHANGE
    IF MASTER-READY-FOR-OUTPUT EQUAL "NO"
        PERFORM NO-MASTER-TO-CHANGE-ERROR
    ELSE
        PERFORM APPLY-TRANS-TO-WS-NEW-MASTER
ELSE IF TRANSACTION-CODE-IS-DELETE
    IF MASTER-READY-FOR-OUTPUT EQUAL "NO"
        PERFORM NO-MASTER-TO-DELETE-ERROR
    ELSE
        MOVE "NO" TO MASTER-READY-FOR-OUTPUT
ELSE
    PERFORM INVALID-TRANS-CODE-ERROR

```

```

PERFORM GET-NEXT-TRANSACTION
DETERMINE-LOW-KEY
  IF TRANSACTION-KEY IS LESS THAN OLD-MASTER-KEY
    MOVE TRANSACTION-KEY TO LOW-KEY
  ELSE
    MOVE OLD-MASTER-KEY TO LOW-KEY

GET-NEXT-TRANSACTION.
  READ TRANSACTION-FILE
  AT END
    MOVE HIGH-VALUES TO TRANSACTION-KEY

GET-NEXT-MASTER.
  READ OLD-MASTER-FILE
  AT END
    MOVE HIGH-VALUES TO OLD-MASTER-KEY

```

شكل (١١ - ٢) تكملة

الخطوة التالية .. هى تحديد إذا ما كان هناك عمليات جارية ، يجب تطبيقها على محتويات WS - NEW - MASTER - RECORD . وطالما أن TRANSACTION - KEY يستمر مساويا للقيمة الحالية LOW - KEY .. تطبق سجلات العمليات الجارية على نفس السجل الرئيسى ويتم إدخالها وتشغيلها . لاحظ أن PERFORM PROCESS - A - TRANSACTION .. UNTIL .. هى التى تسمح للخوارزمى بمعاملة أكثر من سجل عمليات جارية لنفس السجل الرئيسى .

بعد تنفيذ كل العمليات الجارية المناسبة .. يخرج الخوارزمى السجل من WS - NEW - MASTER - RECORD إلى MASTER - FILE (بافتراض وجود سجل نشط فى منطقة البناء) . وعند ذلك .. يتغير المفتاح MASTER - READY - FOR - OUTPUT ليصبح فى وضع OFF مرة أخرى .

أخيرا .. وبعد المدخلات الممكنة لسجل OLD - MASTER - RECORD ، وسجل واحد ، أو أكثر جديد من TRANSAC - TION - RECORD .. ينفذ DETERMINE - LOW - KEY مرة أخرى قبل التكرار UPDATE - MASTER - FILE .

مثال ١١ - ٢ :

يسجل جدول ١٠ . ١١ تطبيق خوارزمى خط الاتزان على ملفات اختبارية :

TRANSACTION-FILE: 15, 20, 20, 60
 OLD-MASTER-FILE: 10, 20, 30, 60, 70

ملاحظات خطوة بخطوة :

١ - يستدعى الخوارزمى بقراءة أول سجل من كل ملف ، وتحديد أقل مفتاح ، وتكون منطقة بناء السجل الرئيسى الجديد فارغة.

٢ - يكون LOW - KEY EQUAL OLD - MASTER - KEY صحيحا ، وعلى هذا ينسخ السجل الرئيسى القديم فى منطقة بناء السجل الرئيسى الجديد ، ويحول المفتاح إلى الحالة on ليحدد أن منطقة العمل تحتوى على سجل ، ويتم إدخال السجل الرئيسى القديم التالى .

٣ - يكون LOW - KEY EQUAL TRANSAION - KEY خطأ ، وعلى هذا .. لا يتم تشغيل أى عمليات جارية . وحيث إن MASTER - READY - FOR - OUTPUT - EQUAL "YES" .. تكتب محتويات منطقة العمل (10) فى الملف الرئيسى الجديد ، ويتحول مفتاح الأعداد الى حالة off .

٤ - يقارن LOW - KEY - DETERMINE الرقم 15 مع الرقم 20 .. فيختار 15 .

٥ - بالنسبة للتنفيذ الثانى لـ UPDATE - MASTER - FILE .. لا يكون LOW - KEY مساويا لـ OLD - MASTER - KEY ، وعلى هذا .. يتم إدخال سجل رئيسى قديم . وحيث إن LOW - KEY مساويا لـ TRANSACTION - KEY .. ينفذ PROCESS - A - TRANSACTION ، لتشغيل العملية الجارية 15 . ولاحظ أنها إذا لم تكن عملية إضافة لعملية جارية .. فإنها تعامل كخطأ ؛ حيث يكون مفتاح التحديد MASTER - READY - NO فى الوضع . بافتراض أنها إضافة .. تنقل العمليات الجارية إلى منطقة العمل ، ويتحول MASTER - READY الى الوضع on .

٦ - يدخل PROCESS - A - TRANSACTION الآن سجل العمليات الجارية التالى .

٧ - حيث إن LOW - KEY لا يساوى TRANSACTION - KEY .. فإن ينفذ PROCESS - A - TRANSACTION مرة أخرى فى هذا الوقت . وبدلا من ذلك يختبر .. الخوارزمى مفتاح MASTER - READY ، والذي يكون فى وضع on ، وعلى هذا .. تكتب 15 فى الملف الرئيسى الجديد ، ويحول المفتاح إلى وضع off .

٨ - يقارن LOW - KEY - DETERMINE الرقم 20 مع 20 .. فيختار 20 .

٩ - عند بدء التنفيذ الثالث لـ UPDATE - MASTER - FILE .. يكون LOW - KEY مساويا لـ OLD - MASTER - KEY ، وعلى هذا .. ينقل السجل الرئيسى القديم إلى منطقة العمل ، ويوضع المفتاح فى وضع on ، ويتم إدخال السجل الرئيسى القديم التالى .

١٠ - حيث إن LOW - KEY يساوى TRANSACTION - KEY .. ينفذ PROCESS - A - TRANSACTION ، متسببا فى تشغيل أول 20 عملية جارية . ولاحظ أنها إذا كانت إضافة .. فنحن نكتشف المفتاح المكرر ، حيث تكون MASTER - READY فى وضع on . بافتراض أنه تغيير .. فإننا نجره على السجل الرئيسى القديم 20 ، فى منطقه العمل .

١١ - يتم إدخال سجل العمليات الجارية التالى .

١٢ - حيث إن LOW - KEY لا يزال مساويا لـ TRANSACTION - KEY .. ينفذ PROCESS - A - TRANSACTION مرة أخرى ، متسببا فى تعديل السجل الرئيسى القديم 20 ، الموجود فى منطقة العمل مرة ثانية (بافتراض أن العملية الجارية هي تغيير) .

Execution of UPDATE-MASTER-FILE	LOW-KEY	Value in TRANSACTION- RECORD	Value in OLD-MASTER- RECORD	Value in WORKING-STORAGE NEW-MASTER Area	Ready Switch
1. first	10	15	10	empty	NO
2. first	10	15	20	10	YES
3. first	10	15	20	empty	NO
4. first	15	15	20	empty	NO
5. second	15	15	20	15	YES
6. second	15	20 (1st)	20	15	YES
7. second	15	20 (1st)	20	empty	NO
8. second	20	20 (1st)	20	empty	NO
9. third	20	20 (1st)	30	20	YES
10. third	20	20 (1st)	30	20 (changed)	YES
11. third	20	20 (2nd)	30	20 (changed)	YES
12. third	20	20 (2nd)	30	20 (changed twice)	YES
13. third	20	60	30	20 (changed twice)	YES
14. third	20	60	30	empty	NO
15. third	30	60	30	empty	NO
16. fourth	30	60	60	30	YES
17. fourth	30	60	60	empty	NO
18. fourth	60	60	60	empty	NO
19. fifth	60	60	70	60	YES
20. fifth	60	60	70	60	NO
21. fifth	60	HIGH-VALUES	70	60	NO
22. fifth	70	HIGH-VALUES	70	60	NO
23. sixth	70	HIGH-VALUES	HIGH-VALUES	70	YES
24. sixth	70	HIGH-VALUES	HIGH-VALUES	empty	NO
25. sixth	HIGH-VALUES	HIGH-VALUES	HIGH-VALUES	empty	NO

جدول (١١ - ١)

- ١٣ - يتم إدخال سجل العمليات الجارية التالى .
- ١٤ - حيث إن LOW - KEY لم يعد مساويا TRANSACTION - KEY .. يتجه الخوارزمى للتأكد من - MASTER READY ، وحيث إنه فى حالة on ، يتم إخراج السجل المعدل الموجود فى منطقة العمل (20) إلى الملف الرئيسى الجديد ، ويحول المفتاح إلى حالة off .
- ١٥ - يقارن LOW - KEY - DETERMINE الرقم 60 مع 30 .. فيختار 30 .
- ١٦ - فى بداية التنفيذ الرابع لـ UPDATE - MASTER - FILE .. يتساوى LOW - KEY مع OLD - MASTER - KEY ، بحيث ينقل السجل الرئيسى القديم إلى منطقة العمل ، ويكون المفتاح فى حالة on ، ويتم إدخال السجل الرئيسى القديم التالى .
- ١٧ - حيث إن LOW - KEY لا يساوى TRANSACTION - KEY .. فلن يجرى أى تشغيل على عمليات جارية ، وحيث إن MASTER - READY فى الوضع on ، فيتم إخراج محتويات منطقة العمل (السجل 30) فى الملف الرئيسى الجديد ، ويتحول MASTER - READY إلى الوضع off .
- ١٨ - يقارن LOW - KEY - TERMINE الرقم 60 مع 60 .. فيختار 60 .
- ١٩ - فى بداية التنفيذ الخامس لـ UPDATE - MASTER - FILE .. يكون LOW - KEY مساويا لـ OLD - MASTER - KEY ، وعلى هذا .. ينقل السجل الرئيسى القديم إلى منطقة العمل ويصبح المفتاح فى وضع on ويتم إدخال السجل الرئيسى القديم التالى .
- ٢٠ - حيث إن LOW - KEY يساوى TRANSACTION - KEY .. يتم تشغيل العملية الجارية الحالية . وبافتراض أنها حذف .. يتحول MASTER - READY إلى وضع off .
- ٢١ - يتم إدخال العملية الجارية التالية ، ويتسبب هذا فى نهاية الملف ، الذى يضع HIGH - VALUES كقيمة لـ TRANS - ACTION - KEY ، وحيث إن LOW - KEY لا يساوى TRANSACTION - KEY .. فلن ينفذ - A - PROCESS TRANSACTION مرة أخرى . وبسبب أن MASTER - READY تحول الى الوضع off عن طريق الحذف .. فلن يتم إخراج محتويات منطقة العمل إلى الملف الرئيسى الجديد .
- ٢٢ - يقارن LOW - KEY - DETERMINE القيمة HIGH - VALUES مع 70 .. فيختار 70 .
- ٢٣ - فى بداية تنفيذ UPDATE - MASTER - FILE للمرة السادسة ، يكون LOW - KEY مساويا لـ OLD - MASTER - KEY وعلى هذا .. ينقل السجل الرئيسى القديم إلى منطقة العمل ، ويكون MASTER - READY فى وضع on ، ويتم ادخال السجل الرئيسى القديم التالى ؛ حيث يتسبب هذا فى نهاية الملف وتحديد قيمة HIGH - VALUES لـ OLD - MASTER - KEY .
- ٢٤ - وحيث إن LOW - KEY لا يساوى TRANSACTION - KEY .. فلن ينفذ - A - PROCESS . وحيث إن قيمة MASTER - READY هى "YES" يتم إخراج محتويات منطقة العمل (السجل 70) فى الملف الرئيسى الجديد ، ويتحول MASTER - READY إلى الوضع off .
- ٢٥ - يقارن LOW - KEY - DETERMINE القيمة HIGH - VALUES مع HIGH - VALUES ؛ مختارا HIGH - VALUES . وحيث إن LOW - KEY لا يتساوى مع HIGH - VALUES .. فلن ينفذ أى تكرار - MAS - UPDATE TER - FILE ، تكون النتيجة هى : الملف الرئيسى الجديد : السجل 10 ، السجل 15 (مضاف) ، السجل 20 (جرى عليه تغيير مرتين) ، السجل 30 ، السجل 70 .

لاحظ حذف السجل 60 .

مثال ١١ - ٣ :

باستخدام خوارزمية خط الاتزان ، أكتب تجديد ملف تتابعي لملف حسابات الدائنين الرئيسي التالي:

```
01 AR-CUSTOMER-MASTER-RECORD.
05 AR-CUSTOMER-ID          PIC X(4).
05 AR-CUSTOMER-NAME        PIC X(20).
05 AR-NUMBER-INVOICES       PIC S9(2)          COMP-3.
05 AR-INVOICE-DATA          OCCURS 1 TO 15 TIMES
                             DEPENDING ON AR-NUMBER-INVOICES
                             ASCENDING KEY IS AR-INVOICE-DATE.
10 AR-INVOICE-ID            PIC X(5).
10 AR-INVOICE-DATE.
15 AR-INVOICE-YY            PIC 99.
15 AR-INVOICE-MM            PIC 99.
15 AR-INVOICE-DD            PIC 99.
10 AR-INVOICE-AMOUNT        PIC S9(5)V99      COMP-3.
```

الملف الرئيسي وملف العمليات الجارية مرتبين طبقاً لحقل ID . وهناك ثلاثة أنواع لسجلات العمليات الجارية .

• حذف سجل عميل من الملف الرئيسي :

customer ID : الأعمدة ١ - ٤

رمز العملية الجارية "1" : العمود ٥

غير مستخدمة : الأعمدة ٦ - ٨٠

• إضافة سجل عميل جديد للملف الرئيسي :

customer ID : الأعمدة ١ - ٤

رمز العملية الجارية "2" : العمود ٥

اسم العميل : الأعمدة ٦ - ٢٥

invoice ID : الأعمدة ٢٦ - ٣٠

invoice date (yyymmdd) : الأعمدة ٣١ - ٣٦

amount + (DISPLAY) : الأعمدة ٣٧ - ٤٢

غير مستخدم : الأعمدة ٤٤ - ٨٨

• تغيير بيانات الفاتورة الموجودة :

Customer ID : الأعمدة ١١ - ٤

رمز الفاتورة ("I" أو "P" أو "A") : العمود ٥

invoice ID : الأعمدة ٧ - ١١

invoice date (yymmdd) :	الأعمدة ١٢ - ١٧
amount [S9(5)V99 DISPLAY] :	الأعمدة ١٨ - ٢٤
غير مستخدم :	الأعمدة ٢٥ - ٨٠

يوجه الرمز "I" الانتباه الى وجود محتوى جديد ، يراد إدخاله فى جدول بيانات الفاتورة (مع حفظ الجدول مرتبا طبقا لتاريخ الفاتورة) . يدعو الرمز "A" إلى استبدال الكمية الموجودة فى الجدول ، بالكمية الموجودة فى سجل العمليات الجارية . يعطى الرمز "1" إشارة بأن هناك مبلغاً مدفوعاً بقيمة الفاتورة . إذا كانت قيمة الفاتورة صفراً بعد طرح هذا المبلغ .. تحذف محتويات بيانات الفاتورة واقعيها ، إما إذا كان سالبا .. يسرى الحذف مع كتابة سجل فى ملف تتابعى اسمه - CREDIT FILE ، وتوجد فيه الحقول - CREDIT - BAL , CUSTOMER - ID , CUSTOMER - NAME , INVOICE - ID ، وتوجد فى الملف - INVOICE - DATE (yymmdd) , ANCE [S9 (5)V99COMP - 3] . إذا ترك سجل رئيسى بدون فواتير ، .. فإنه يحذف من الملف . بالإضافة إلى CREDIT - FILE , NEW - AR - CUSTOMER - MASTER - FILE . يتم إخراج ERROR REPORT - ، يشمل عناوين للصفحة (مع رقم الصفحة وتاريخ التنفيذ) لكل صفحة ، وكل حقول العمليات الجارية لكل سجل خطأ وعبرة باللغة الإنجليزية تصف الخطأ .. يجب أن تغطى الأخطاء التالية :

- (١) رمز عملية جارية 1 ، مع عدم وجود customer ID فى الملف .
 - (٢) رمز عملية جارية 2 ، مع وجود customer ID فعلا فى الملف .
 - (٣) رمز عملية جارية 3 ، مع عدم وجود customer ID فى الملف .
 - (٤) رمز العملية الجارية 3I ، مع وجود invoice ID فعلا فى جدول الفواتير .
 - (٥) رمز العملية 3A ، مع عدم وجود invoice ID فى جدول الفواتير .
 - (٦) رمز العملية 3P ، مع عدم وجود invoice ID فى جدول الفواتير ، أو مع عدم اتفاق تاريخ الفاتورة ، مع تاريخ الفاتورة فى الملف الرئيسى .
 - (٧) لا يكون تاريخ فاتورة العملية الجارية أو كمية الفاتورة عدديا .
- يوجد أحد الحلول فى شكل (١١ - ٣) فيما يلى .

```

00001 IDENTIFICATION DIVISION.
00002
00003 PROGRAM-ID. SEQUUPDAT.
00004
00005 AUTHOR. LARRY NEWCOMER.
00006 INSTALLATION. PENN STATE UNIVERSITY, YORK CAMPUS.
00007
00008 DATE-WRITTEN. MAY 1983.
00009 DATE-COMPILED. MAY 9, 1983.
00010 SECURITY. NONE.
00011
00012 * SEQUENTIAL FILE UPDATE USING THE BALANCED LINE
00013 * ALGORITHM. INPUT: OLD MASTER FILE
00014

```

شكل (١١ - ٣)

```

00015      *      TRANSACTION FILE
00016      *      OUTPUT: NEW MASTER FILE
00017      *      ERROR REPORT
00018      *      CREDIT FILE

00020      ENVIRONMENT DIVISION.

00022      CONFIGURATION SECTION.
00023      SOURCE-COMPUTER.  IBM-3081.
00024      OBJECT-COMPUTER. IBM-3081.

00026      INPUT-OUTPUT SECTION.

00028      FILE-CONTROL.
00029
00030      SELECT OLD-MASTER-FILE
00031      ASSIGN TO OLDMAST
00032      ORGANIZATION IS SEQUENTIAL
00033      ACCESS IS SEQUENTIAL
00034      .
00035
00036      SELECT TRANSACTION-FILE
00037      ASSIGN TO TRANS
00038      ORGANIZATION IS SEQUENTIAL
00039      ACCESS IS SEQUENTIAL
00040
00041
00042      SELECT NEW-MASTER-FILE
00043      ASSIGN TO NEWMAST
00044      ORGANIZATION IS SEQUENTIAL
00045      ACCESS IS SEQUENTIAL
00046      .
00047
00048      SELECT CREDIT-FILE
00049      ASSIGN TO CREDITS
00050      ORGANIZATION IS SEQUENTIAL
00051      ACCESS IS SEQUENTIAL
00052      .
00053
00054      SELECT ERROR-REPORT
00055      ASSIGN TO ERRORLOG
00056      ORGANIZATION IS SEQUENTIAL
00057      ACCESS IS SEQUENTIAL
00058
00059

00061      DATA DIVISION.

00063      FILE SECTION.
00064
00065      *
00066      * -----
00067      *
00068      *      USE OF VARIABLE-LENGTH RECORDS TO SAVE DISK/TAPE SPACE
00069      *
00070      * -----
00071      *
00072      FD  OLD-MASTER-FILE
00073      BLOCK CONTAINS 0 RECORDS

```

```

00074      RECORD CONTAINS 41 TO 251 CHARACTERS
00075      LABEL RECORDS ARE STANDARD
00076
00077
00078      01  OM-AR-CUSTOMER-RECORD.
00079          05  OM-AR-CUSTOMER-ID          PIC X(4).
00080          05  OM-AR-CUSTOMER-NAME        PIC X(20).
00081          05  OM-AR-NUMBER-INVOICES      PIC S9(2)      COMP-3.
00082
00083      * -----
00084      *
00085      *  VARIABLE-LENGTH TABLE DEFINED WITHIN LOGICAL RECORD
00086      * -----
00087      *
00088      *
00089          05  OM-AR-INVOICE-DATA          OCCURS 1 TO 15 TIMES
00090                                  DEPENDING ON
00091                                  OM-AR-NUMBER-INVOICES
00092
00093              10  OM-INVOICE-ID          PIC X(5).
00094              10  OM-INVOICE-DATE.
00095                  15  OM-INVOICE-YY      PIC 99.
00096                  15  OM-INVOICE-MM      PIC 99.
00097                  15  OM-INVOICE-DD      PIC 99.
00098              10  OM-INVOICE-AMOUNT      PIC S9(5)V99  COMP-3.
00099
00100      FD  TRANSACTION-FILE
00101          RECORD CONTAINS 80 CHARACTERS
00102          LABEL RECORDS ARE OMITTED
00103
00104
00105      01  TRANSACTION-RECORD.
00106          05  TRANS-SORT-KEY.
00107              10  TRANS-CUSTOMER-ID      PIC X(4).
00108              10  TRANS-CODE              PIC X.
00109                  88  DELETION-TRANSACTION      VALUE "1".
00110                  88  ADD-TRANSACTION          VALUE "2".
00111                  88  CHANGE-TRANSACTION        VALUE "3".
00112          05  TRANS-VARIABLE-AREA        PIC X(75).
00113
00114      * -----
00115      *
00116      *  USE OF REDEFINES TO DESCRIBE THE VARIOUS FORMS OF
00117      *  TRANSACTION RECORDS
00118      * -----
00119      *
00120      *
00121          05  TRANS-ADD-AREA              REDEFINES TRANS-VARIABLE-AREA.
00122              10  ADD-CUSTOMER-NAME      PIC X(20).
00123              10  ADD-INVOICE-ID         PIC X(5).
00124              10  ADD-INVOICE-DATE       PIC 9(6).
00125              10  ADD-AMOUNT             PIC S9(5)V99.
00126              10  FILLER                 PIC X(37).
00127          05  TRANS-CHANGE-AREA          REDEFINES TRANS-VARIABLE-AREA.
00128              10  TRANS-INVOICE-CODE     PIC X.
00129                  88  INSERTION-CODE      VALUE "I".
00130                  88  ADJUSTMENT-CODE     VALUE "A".

```

```

00131          88 PAYMENT-CODE                      VALUE "P".
00132          10 CHANGE-INVOICE-ID PIC X(5).
00133          10 CHANGE-INVOICE-DATE PIC X(6).
00134          10 CHANGE-AMOUNT PIC S9(5)V99.
00135          10 CHANGE-AMOUNT-X REDEFINES CHANGE-AMOUNT
00136                               PIC X(7).
00137          10 FILLER PIC X(56).
00138
00139
00140      FD NEW-MASTER-FILE
00141          BLOCK CONTAINS 0 RECORDS
00142
00143          RECORD CONTAINS 41 TO 251 CHARACTERS
00144          LABEL RECORDS ARE STANDARD
00145
00146      01 NM-AR-CUSTOMER-RECORD.
00147          05 NM-AR-CUSTOMER-ID PIC X(4).
00148          05 NM-AR-CUSTOMER-NAME PIC X(20).
00149          05 NM-AR-NUMBER-INVOICES PIC S9(2) COMP-3.
00150          05 NM-AR-INVOICE-DATA OCCURS 1 TO 15 TIMES
00151                               DEPENDING ON
00152                               NM-AR-NUMBER-INVOICES
00153
00154          10 NM-INVOICE-ID PIC X(5).
00155          10 NM-INVOICE-DATE.
00156              15 NM-INVOICE-YY PIC 99.
00157              15 NM-INVOICE-MM PIC 99.
00158              15 NM-INVOICE-DD PIC 99.
00159          10 NM-INVOICE-AMOUNT PIC S9(5)V99 COMP-3.
00160
00161      FD CREDIT-FILE
00162          BLOCK CONTAINS 0 RECORDS
00163          RECORD CONTAINS 39 CHARACTERS
00164          LABEL RECORDS ARE STANDARD
00165
00166      01 CREDIT-BALANCE-RECORD.
00167          05 CREDIT-CUSTOMER-ID PIC X(4).
00168          05 CREDIT-CUSTOMER-NAME PIC X(20).
00169          05 CREDIT-INVOICE-ID PIC X(5).
00170          05 CREDIT-INVOICE-DATE PIC 9(6).
00171          05 CREDIT-BALANCE PIC 9(5)V99 COMP-3.
00172
00173      FD ERROR-REPORT
00174          RECORD CONTAINS 132 CHARACTERS
00175          LABEL RECORDS ARE OMITTED
00176          LINAGE IS 60
00177              WITH FOOTING AT 59
00178              LINES AT TOP 3
00179              LINES AT BOTTOM 3
00180
00181
00182
00183      01 ERROR-REPORT-LINE PIC X(132).
00184
00185      WORKING-STORAGE SECTION.
00186
00187      01 WS-SWITCHES-AND-WORK-AREAS.
00188          05 WS-PAGE-NUMBER PIC S9(3) COMP-3.
00189          05 LOW-KEY PIC X(4).

```

شكل (١١ - ٣) تكملة

```

00190      05 MAXIMUM-TABLE-SIZE      PIC S9(3)      COMP-3
00191                                         VALUE +15.
00192      05 MASTER-READY              PIC X(3).
00193          88 MASTER-RECORD-PRESENT      VALUE "YES".
00194          88 NO-MASTER-RECORD          VALUE "NO ".
00195      05 SYSTEM-DATE.
00196          10 SYSTEM-YY              PIC 99.
00197          10 SYSTEM-MM              PIC 99.
00198          10 SYSTEM-DD              PIC 99.
00199      05 WS-DIFFERENCE              PIC S9(5)V99    COMP-3.
00200
00201      *
00202      * -----
00203      *
00204      * ERROR MESSAGE TEXT DEFINED AS PROGRAM CONSTANTS RATHER
00205      * THAN USING NON-NUMERIC LITERALS IN PROCEDURE DIVISION
00206      *
00207      * -----
00208      *
00209      01 WS-ERROR-MESSAGE-AREAS.
00210          05 ADD-ERROR-MESSAGE        PIC X(42).
00211                                         VALUE "DUPLICATE KEY--NO ADD".
00212          05 CHANGE-ERROR-MESSAGE     PIC X(42)
00213                                         VALUE "KEY NOT FOUND--NO CHANGE".
00214          05 DELETION-ERROR-MESSAGE    PIC X(42)
00215                                         VALUE "KEY NOT FOUND--NO DELETE".
00216          05 INVALID-TRANS-CODE-MESSAGE
00217                                         PIC X(42)
00218                                         VALUE "INVALID CODE--NO ACTION".
00219          05 INVALID-CHANGE-CODE-MESSAGE
00220                                         PIC X(42)
00221                                         VALUE "INVALID CHANGE--IGNORED".
00222          05 ADJUST-ERROR-MESSAGE      PIC X(42)
00223                                         VALUE "INVOICE NOT FOUND".
00224          05 INVALID-FIELDS-MESSAGE    PIC X(42)
00225                                         VALUE "INVALID AMOUNT/DATE".
00226          05 NO-PAYMENT-INVOICE-MESSAGE
00227                                         PIC X(42)
00228                                         VALUE "NO INVOICE--PAYMENT NOT CREDITED".
00229          05 DUPLICATE-INVOICE-MESSAGE
00230                                         PIC X(42)
00231                                         VALUE "DUPLICATE INVOICE--IGNORED".
00232          05 TABLE-OVERFLOW-MESSAGE   PIC X(42)
00233                                         VALUE ">15 INVOICES--IGNORED".
00234
00235      *
00236      * -----
00237      *
00238      * WORKING-STORAGE AREA USED IN BALANCED LINE ALGORITHM
00239      * TO CONSTRUCT NEW-MASTER-FILE RECORD WITH ALL
00240      * TRANSACTIONS APPLIED
00241      *
00242      * -----
00243      *
00244      01 WS-AR-CUSTOMER-RECORD.
00245          05 WS-AR-CUSTOMER-ID          PIC X(4).
00246          05 WS-AR-CUSTOMER-NAME        PIC X(20).
00247          05 WS-AR-NUMBER-INVOICES      PIC S9(2)      COMP-3.
00248          05 WS-AR-INVOICE-DATA         OCCURS 1 TO 15 TIMES

```

```

00249                                     DEPENDING ON
00250                                     WS-AR-NUMBER-INVOICES
00251                                     ASCENDING KEY IS
00252                                     WS-INVOICE-DATE
00253                                     INDEXED BY WS-INDEX
00254                                     WS-INDEX-2
00255
00256                                     10 WS-INVOICE-ID          PIC X(5).
00257                                     10 WS-INVOICE-DATE.
00258                                     15 WS-INVOICE-YY        PIC 99.
00259                                     15 WS-INVOICE-MM        PIC 99.
00260                                     15 WS-INVOICE-DD        PIC 99.
00261                                     10 WS-INVOICE-AMOUNT     PIC S9(5)V99  COMP-3.
00262
00263
00264 01 WS-ERROR-PAGE-TITLE.
00265     05 FILLER          PIC X(20)      VALUE SPACES.
00266     05 FILLER          PIC X(25)
00267                          VALUE "TRANSACTION ERROR LOG".
00268     05 FILLER          PIC X(5)       VALUE "PAGE ".
00269     05 WS-ERROR-PAGE-NUMBER PIC Z9.
00270     05 FILLER          PIC XX        VALUE SPACES.
00271     05 WS-ERROR-PAGE-DATE.
00272         10 WS-ERROR-MM    PIC Z9.
00273         10 FILLER         PIC X       VALUE "/".
00274         10 WS-ERROR-DD    PIC 99.
00275         10 FILLER         PIC X       VALUE "/".
00276
00276         10 WS-ERROR-YY    PIC 99.
00277     05 FILLER          PIC X(69)     VALUE SPACES.
00278
00279 01 WS-ERROR-HEADING.
00280     05 FILLER          PIC X(9)      VALUE "CUSTOMER"
00281     05 FILLER          PIC X(5)      VALUE "CODE".
00282     05 FILLER          PIC X(118)
00283                          VALUE "REST OF TRANSACTION".
00284
00285 01 WS-ERROR-LINE.
00286     05 WS-ERROR-CUSTOMER-ID PIC X(4).
00287     05 FILLER          PIC X(5)     VALUE SPACES.
00288     05 WS-ERROR-CODE      PIC X.
00289     05 FILLER          PIC X(4)     VALUE SPACES.
00290     05 WS-ERROR-REST-OF-TRAN PIC X(75).
00291     05 FILLER          PIC X(1)     VALUE SPACES.
00292     05 WS-ERROR-MESSAGE   PIC X(42).
00293
00295     PROCEDURE DIVISION.
00297
00298     * -----
00299     *
00300     * COBOL IMPLEMENTATION OF BALANCED LINE ALGORITHM*
00301     * FOR SEQUENTIAL FILE UPDATE
00302     *
00303     * -----
00304     *
00305     000-SEQUENTIAL-FILE-UPDATE.
00306

```

```

00307      MOVE "NO" TO MASTER-READY
00308      MOVE ZERO TO WS-PAGE-NUMBER
00309      ACCEPT SYSTEM-DATE FROM DATE
00310      MOVE SYSTEM-YY TO WS-ERROR-YY
00311      MOVE SYSTEM-MM TO WS-ERROR-MM
00312      MOVE SYSTEM-DD TO WS-ERROR-DD
00313      OPEN      INPUT              OLD-MASTER-FILE
00314                                TRANSACTION-FILE
00315                                NEW-MASTER-FILE
00316                                CREDIT-FILE
00317                                ERROR-REPORT
00318      PERFORM 170-PRINT-ERROR-REPORT-HEADING
00319      PERFORM 030-READ-TRANSACTION
00320      PERFORM 040-READ-MASTER
00321      PERFORM 050-DETERMINE-LOW-KEY
00322      PERFORM 010-UPDATE-MASTER-FILE
00323      UNTIL LOW-KEY EQUAL HIGH-VALUES
00324
00325      CLOSE   OLD-MASTER-FILE
00326            TRANSACTION-FILE
00327            NEW-MASTER-FILE
00328            CREDIT-FILE
00329            ERROR-REPORT
00330      STOP RUN
00331
00332
00333      010-UPDATE-MASTER-FILE.
00334
00335      IF LOW-KEY EQUAL OM-AR-CUSTOMER-ID
00336      *
00337      * -----
00338      *
00339      * ILLUSTRATES MOVING VARIABLE-LENGTH TABLE
00340      *
00341      * -----
00342      *
00343
00344      MOVE OM-AR-NUMBER-INVOICES TO WS-AR-NUMBER-INVOICES
00345      MOVE OM-AR-CUSTOMER-RECORD TO WS-AR-CUSTOMER-RECORD
00346      MOVE "YES" TO MASTER-READY
00347      PERFORM 040-READ-MASTER
00348
00349      IF LOW-KEY EQUAL TRANS-CUSTOMER-ID
00350      PERFORM 020-PROCESS-TRANSACTIONS
00351      UNTIL LOW-KEY NOT EQUAL TRANS-CUSTOMER-ID
00352
00353      IF MASTER-RECORD-PRESENT
00354      PERFORM 060-WRITE-NEW-MASTER-RECORD
00355      MOVE "NO" TO MASTER-READY
00356
00357      PERFORM 050-DETERMINE-LOW-KEY
00358
00359      020-PROCESS-TRANSACTIONS.
00360
00361      *
00362      * -----
00363      *
00364      * NOTE THAT THIS VERSION OF THE BALANCED LINE ALGORITHM
00365      * VALIDATES ALL TRANSACTIONS BEFORE APPLYING THEM
00366      *
00367      * -----
00368      *

```

شكل (١١ - ٣) تكملة


```

00369      IF ADD-TRANSACTION
00370          IF MASTER-RECORD-PRESENT
00371              MOVE ADD-ERROR-MESSAGE TO WS-ERROR-MESSAGE
00372              PERFORM 070-WRITE-ERROR-LINE
00373          ELSE
00374              IF ADD-INVOICE-DATE NUMERIC AND ADD-AMOUNT NUMERIC
00375                  PERFORM 080-MOVE-TRANS-TO-MASTER-AREA
00376                  MOVE "YES" TO MASTER-READY
00377              ELSE
00378                  MOVE INVALID-FIELDS-MESSAGE TO WS-ERROR-MESSAGE
00379                  PERFORM 070-WRITE-ERROR-LINE
00380          ELSE IF CHANGE-TRANSACTION
00381              IF NO-MASTER-RECORD
00382                  MOVE CHANGE-ERROR-MESSAGE TO WS-ERROR-MESSAGE
00383                  PERFORM 070-WRITE-ERROR-LINE
00384              ELSE
00385                  IF CHANGE-INVOICE-DATE NUMERIC
00386                      AND CHANGE-AMOUNT NUMERIC
00387                      PERFORM APPLY-TRANSACTION-TO-MASTER
00388              ELSE
00389                  MOVE INVALID-FIELDS-MESSAGE TO WS-ERROR-MESSAGE
00390                  PERFORM 070-WRITE-ERROR-LINE
00391          ELSE IF DELETION-TRANSACTION
00392              IF NO-MASTER-RECORD
00393                  MOVE DELETION-ERROR-MESSAGE TO WS-ERROR-MESSAGE
00394                  PERFORM 070-WRITE-ERROR-LINE
00395              ELSE
00396                  MOVE "NO" TO MASTER-READY
00397          ELSE
00398              MOVE INVALID-TRANS-CODE-MESSAGE TO WS-ERROR-MESSAGE
00399              PERFORM 070-WRITE-ERROR-LINE
00400      PERFORM 030-READ-TRANSACTION
00401
00402      030-READ-TRANSACTION.
00403
00404      READ TRANSACTION-FILE
00405      AT END
00406          MOVE HIGH-VALUES TO TRANS-CUSTOMER-ID
00407
00408      040-READ-MASTER.
00409
00410      READ OLD-MASTER-FILE
00411      AT END
00412          MOVE HIGH-VALUES TO OM-AR-CUSTOMER-ID
00413
00414      050-DETERMINE-LOW-KEY.
00415
00416      IF TRANS-CUSTOMER-ID IS LESS THAN OM-AR-CUSTOMER-ID
00417          MOVE TRANS-CUSTOMER-ID TO LOW-KEY
00418      ELSE
00419          MOVE OM-AR-CUSTOMER-ID TO LOW-KEY
00420
00421      060-WRITE-NEW-MASTER-RECORD.
00422
00423
00424
00425
00426
00427

```

```

00428 *
00429 *
00430 *
00431 * CHECK FOR VARIABLE-LENGTH DEPENDING ON ITEM = ZERO
00432 * BEFORE WRITING LOGICAL RECORD TO FILE
00433 *
00434 *
00435 *
00436 * IF WS-AR-NUMBER-INVOICES POSITIVE
00437 *     MOVE WS-AR-NUMBER-INVOICES TO NM-AR-NUMBER-INVOICES
00438 *     MOVE WS-AR-CUSTOMER-RECORD TO NM-AR-CUSTOMER-RECORD
00439 *     WRITE NM-AR-CUSTOMER-RECORD
00440 *
00441 *
00442 * 070-WRITE-ERROR-LINE.
00443 *
00444 *     MOVE TRANS-CUSTOMER-ID TO WS-ERROR-CUSTOMER-ID
00445 *     MOVE TRANS-CODE TO WS-ERROR-CODE
00446 *     MOVE TRANS-VARIABLE-AREA TO WS-ERROR-REST-OF-TRAN
00447 *     WRITE ERROR-REPORT-LINE
00448 *     FROM WS-ERROR-LINE
00449 *     AFTER ADVANCING 2 LINES
00450 *     AT END-OF-PAGE
00451 *     PERFORM 170-PRINT-ERROR-REPORT-HEADING
00452 *
00453 *
00454 * 080-MOVE-TRANS-TO-MASTER-AREA.
00455 *
00456 *     MOVE TRANS-CUSTOMER-ID TO WS-AR-CUSTOMER-ID
00457 *     MOVE ADD-CUSTOMER-NAME TO WS-AR-CUSTOMER-NAME
00458 *     MOVE 1 TO WS-AR-NUMBER-INVOICES
00459 *     MOVE ADD-INVOICE-ID TO WS-INVOICE-ID (1)
00460 *     MOVE ADD-INVOICE-DATE TO WS-INVOICE-DATE (1)
00461 *     MOVE ADD-AMOUNT TO WS-INVOICE-AMOUNT (1)
00462 *
00463 *
00464 * APPLY-TRANSACTION-TO-MASTER.
00465 *
00466 *     IF INSERTION-CODE
00467 *         PERFORM 090-INSERT-NEW-INVOICE
00468 *     ELSE IF ADJUSTMENT-CODE
00469 *         PERFORM 130-ADJUST-AN-INVOICE
00470 *     ELSE IF PAYMENT-CODE
00471 *         PERFORM 140-APPLY-PAYMENT
00472 *     ELSE
00473 *         MOVE INVALID-CHANGE-CODE-MESSAGE TO WS-ERROR-MESSAGE
00474 *         PERFORM 070-WRITE-ERROR-LINE
00475 *
00476 *
00477 * 090-INSERT-NEW-INVOICE.
00478 *
00479 *
00480 *
00481 *
00482 * CHECK FOR TABLE OVERFLOW BEFORE INSERTING NEW INVOICE
00483 *
00484 *
00485 *

```

```

00486 IF WS-AR-NUMBER-INVOICES EQUAL MAXIMUM-TABLE-SIZE
00487     MOVE TABLE-OVERFLOW-MESSAGE TO WS-ERROR-MESSAGE
00488     PERFORM 070-WRITE-ERROR-LINE
00489 ELSE
00490 *
00491 * -----
00492 *
00493 * ILLUSTRATES USE OF SEQUENTIAL SEARCH
00494 * -----
00495 *
00496 *
00497     SET WS-INDEX TO 1
00498     SEARCH WS-AR-INVOICE-DATA
00499     AT END
00500     PERFORM 100-LOCATE-AND-MOVE
00501     WHEN WS-INVOICE-ID (WS-INDEX) EQUAL CHANGE-INVOICE-ID
00502     MOVE DUPLICATE-INVOICE-MESSAGE TO WS-ERROR-MESSAGE
00503     PERFORM 070-WRITE-ERROR-LINE
00504
00505
00506 100-LOCATE-AND-MOVE.
00507
00508     SET WS-INDEX TO 1
00509     SEARCH WS-AR-INVOICE-DATA
00510     AT END
00511     ADD 1 TO WS-AR-NUMBER-INVOICES
00512     PERFORM 120-MOVE-IN-NEW-ENTRY
00513     WHEN WS-INVOICE-DATE (WS-INDEX) GREATER THAN
00514     CHANGE-INVOICE-DATE
00515     ADD 1 TO WS-AR-NUMBER-INVOICES
00516     PERFORM 110-MOVE-ENTRIES-DOWN
00517     VARYING WS-INDEX-2 FROM WS-AR-NUMBER-INVOICES
00518     BY -1
00519     UNTIL WS-INDEX-2 EQUAL WS-INDEX
00520     PERFORM 120-MOVE-IN-NEW-ENTRY
00521
00522
00523 110-MOVE-ENTRIES-DOWN.
00524
00525     MOVE WS-AR-INVOICE-DATA (WS-INDEX-2 - 1) TO
00526     WS-AR-INVOICE-DATA (WS-INDEX-2)
00527
00528
00529 120-MOVE-IN-NEW-ENTRY.
00530
00531     MOVE CHANGE-INVOICE-ID TO WS-INVOICE-ID (WS-INDEX)
00532     MOVE CHANGE-INVOICE-DATE TO WS-INVOICE-DATE (WS-INDEX)
00533     MOVE CHANGE-AMOUNT TO WS-INVOICE-AMOUNT (WS-INDEX)
00534
00535
00536 130-ADJUST-AN-INVOICE.
00537
00538     SET WS-INDEX TO 1
00539     SEARCH WS-AR-INVOICE-DATA
00540     AT END
00541     MOVE ADJUST-ERROR-MESSAGE TO WS-ERROR-MESSAGE
00542     PERFORM 070-WRITE-ERROR-LINE
00543     WHEN WS-INVOICE-ID (WS-INDEX) EQUAL CHANGE-INVOICE-ID

```

```

00544             MOVE CHANGE-AMOUNT TO WS-INVOICE-AMOUNT (WS-INDEX)
00545
00546
00547     140-APPLY-PAYMENT.
00548
00549             SET WS-INDEX TO 1
00550             SEARCH WS-AR-INVOICE-DATA
00551                 AT END
00552                 MOVE NO-PAYMENT-INVOICE-MESSAGE TO WS-ERROR-MESSAGE
00553                 PERFORM 070-WRITE-ERROR-LINE
00554                 WHEN WS-INVOICE-ID (WS-INDEX) EQUAL CHANGE-INVOICE-ID
00555                 PERFORM 150-CALCULATE-BALANCE
00556
00557
00558     150-CALCULATE-BALANCE.
00559
00560             SUBTRACT CHANGE-AMOUNT FROM WS-INVOICE-AMOUNT (WS-INDEX)
00561             GIVING WS-INVOICE-AMOUNT (WS-INDEX)
00562             WS-DIFFERENCE
00563 *
00564 * -----
00565 *
00566 *     IF OVERPAID, WRITE CUSTOMER INFORMATION TO CREDIT FILE
00567 *
00568 * -----
00569 *
00570             IF WS-DIFFERENCE NEGATIVE
00571                 MOVE TRANS-CUSTOMER-ID          TO CREDIT-CUSTOMER-ID
00572                 MOVE WS-AR-CUSTOMER-NAME        TO CREDIT-CUSTOMER-NAME
00573                 MOVE CHANGE-INVOICE-ID          TO CREDIT-INVOICE-ID
00574                 MOVE CHANGE-INVOICE-DATE        TO CREDIT-INVOICE-DATE
00575                 MOVE WS-DIFFERENCE              TO CREDIT-BALANCE
00576             * WRITE CREDIT-BALANCE-RECORD
00577
00578 *
00579 * -----
00580 *
00581 *     IF OVERPAID OR PAID OUT, REMOVE INVOICE FROM TABLE
00582 *
00583 * -----
00584 *
00585             IF WS-DIFFERENCE NOT POSITIVE
00586             PERFORM 160-REMOVE-INVOICE
00587                 VARYING WS-INDEX FROM WS-INDEX BY 1
00588                 UNTIL WS-INDEX GREATER THAN WS-AR-NUMBER-INVOICES
00589             SUBTRACT 1 FROM WS-AR-NUMBER-INVOICES
00590
00591
00592     160-REMOVE-INVOICE.
00593
00594             MOVE WS-AR-INVOICE-DATA (WS-INDEX + 1) TO
00595             WS-AR-INVOICE-DATA (WS-INDEX)
00596
00597
00598     170-PRINT-ERROR-REPORT-HEADING.
00599
00600             ADD 1 TO WS-PAGE-NUMBER
00601             MOVE WS-PAGE-NUMBER TO WS-ERROR-PAGE-NUMBER
00602             WRITE ERROR-REPORT-LINE

```

شكل (١١ - ٣) تكملة

```

00603      FROM WS-ERROR-PAGE-TITLE
00604      AFTER ADVANCING PAGE
00605      WRITE ERROR-REPORT-LINE
00606      FROM WS-ERROR-HEADING
00607      AFTER ADVANCING 2 LINES
00608

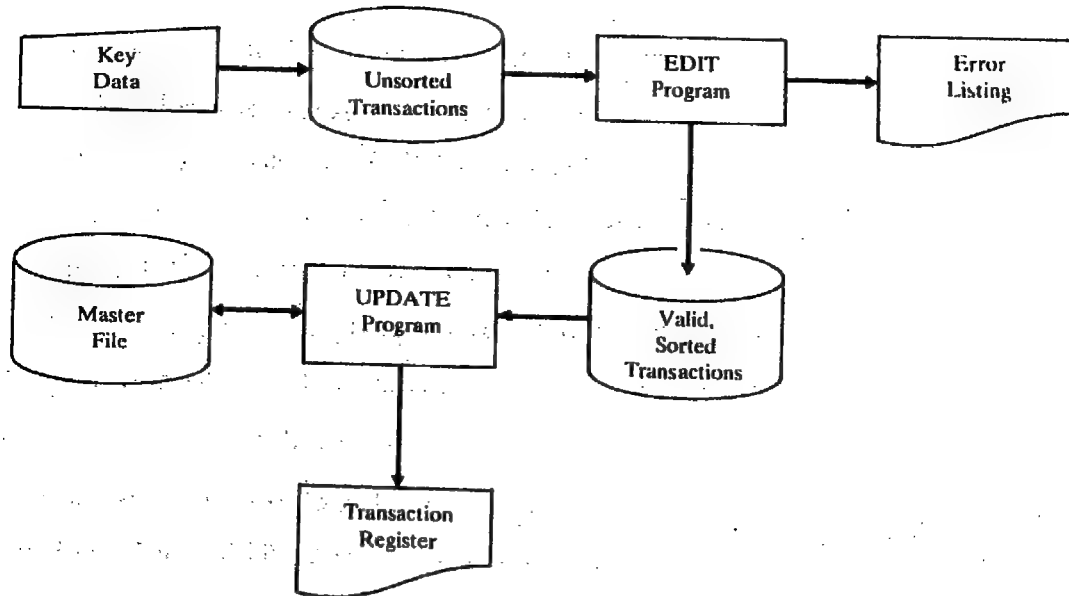
```

شكل (١١ - ٣) تكملة

١١ - ٣ التجديد فى نفس المكان

عندما تخزن ملفات رئيسية مرتبة تتابعيا على قرص .. فمن الممكن تجديد السجلات المنطقية ، دون إنتاج ملف رئيسى جديد : ويحدث هذا عن طريق قراءة السجلات فى الذاكرة الرئيسية ، وإجراء التغييرات عليها ، ثم إعادة كتابتها فى نفس مواقعها على القرص بالضبط . انظر إلى خريطة مسار النظام فى شكل (١١ - ٤) . تقع القيود التالية على هذه العملية :

- (١) لا تضاف سجلات إلى الملف .
- (٢) يجب أن يحدث الحذف منطقيا (يعمل إشارة على السجلات التى لها رمز حذف) .
- (٣) يجب ألا يتغير طول السجل المنطقى .
- (٤) يجب أن يفتح الملف كملف مدخلات ومخرجات I-O (انظر الفصل السادس) .



شكل (١١ - ٤)

يستخدم فعل REWRITE فى إخراج سجلات منطقية ، على نفس موقع القرص الذى سبق قراءته منه ، ويأخذ الشكل

التالى :

REWRITE record-name [FROM identifier]

مثال ١١ - ٤ :

FD MASTER-FILE
01 MASTER-RECORD...

.....
OPEN I-O MASTER-FILE
READ MASTER-FILE AT END...
PERFORM MAKE-CHANGES-TO-MASTER-RECORD
REWRITE MASTER-RECORD

لاحظ أنه لى يجرى تجديداً فى نفس المكان .. فإنه يجب أن يفتح الملف على أنه ملف مدخلات ومخرجات I-O ، ويسمح ذلك بأن تعمل عبارة READ كمداخلات معتادة ، وعبارة REWRITE تستخدم فى إخراج السجل المنطقى فى نفس موقعه الواقعى من القرص (نفس الموقع الذى استخدمته آخر READ فى القراءة) .

أسئلة مراجعة

- ١١- ١ ما (أ) إنتاج ملف . (ب) استرجاع ملف . (ج) صيانة (تجديد) ملف .
- ١١- ٢ ناقش العمليات الثلاث التى يجب أن تؤدى كجزء من الصيانة .
- ١١- ٣ ما الغرض من منطقة FILE STATUS ؟ كيف تعرف وكيف تستخدم ؟
- ١١- ٤ عرف : (أ) الملف الرئيسى (ب) ملف العمليات الجارية (ج) حقل رئيسى (حقل مفتاح) .
- ١١- ٥ اذكر تسلسل الخطوات المتبعة فى تجديد ملف تتابعى تقليدى .
- ١١- ٦ وضح اصطلاح الملف الرئيسى القديم ، واصطلاح الملف الرئيسى الجديد .
- ١١- ٧ ناقش طريقة الأجداد والآباء والأطفال للاحتياطى .
- ١١- ٨ ما المقصود بمسجل العمليات الجارية ؟
- ١١- ٩ ارسم خريطة مسار نظام لتجديد ملف تتابعى تقليدى .
- ١١- ١٠ وضح خوارزمية خط الاتزان ، مستخدماً شجرة شبيهة .
- ١١- ١١ ماذا يعنى التجديد فى نفس المكان ؟
- ١١- ١٢ قارن فتح الملف فى : (أ) INPUT (ب) OUTPUT (ج) I-O .
- ١١- ١٣ ما القيد الذى يجب وضعها على تجديد الملف التتابعى فى نفس المكان ؟
- ١١- ١٤ وضح استخدام فعل REWRITE .
- ١١- ١٥ ارسم خريطة مسار نظام لتجديد ملف تتابعى فى نفس المكان .
- ١١- ١٦ اذكر إحدى مميزات وإحدى عيوب التجديد فى نفس المكان ، بالمقارنة بالتجديد التقليدى (ملف رئيسى قديم ، ولف رئيسى جديد) . (ملاحظة : اعتبر أنه لن يجرى تغيير على السجلات وخذ فى الاعتبار عملية الاحتياطى) .

مسائل محلولة

١١ - ١٧ ما الفرق بين تجديد الملف وتجديد السجل ؟

يشير تجديد الملف إلى صيانة الملف ، والذي تحدث فيه إضافة سجلات الملف ، أو حذف سجلات من الملف ، أو تغيير فى محتويات سجلات الملف . ويشير تجديد السجل إلى عملية تغيير السجلات المنطقية أثناء تجديد الملف .

١١ - ١٨ حدد الخطأ فى البيانات الاختبارية لتجديد الملف التالى :

TRANSACTION-FILE: 4, 23, 15, 12, 30, 50
MASTER-FILE: 10, 20, 30, 40, 50, 60, 70

يجب أن يكون ملف العمليات الجارية مرتباً بنفس الحقل المستخدم فى ترتيب الملف الرئيسى . ويجب أن تحتوى سجلات العمليات الجارية على الحقل الرئيسى (المفتاح) - المرتبة على أساسه سجلات الملف الرئيسى المراد تجديده .

١١ - ١٩ بالإشارة الى السؤال الثانى عشر المذكور عليه ، اذكر بدلاً آخر لعبارة OPEN .

OPEN EXTEND ؛ حيث يمكن إضافة سجلات جديدة الى نهاية الملف ، باستخدام فعل WRITE .

١١ - ٢٠ كم عدد أجيال ملفات العمليات الجارية ، والملفات الرئيسة القديمة التى يجب حفظها ، ولأى فترة زمنية ، عند استخدام احتياطي الأجداد والآباء والأطفال ؟

لا توجد إجابة واحدة : فاهمية التطبيق والبيانات لجال الأعمال ، والمتطلبات القانونية التى تحددها الحكومة ، ومتطلبات الحسابات وما إلى ذلك ، يجب أن تؤخذ كلها فى الحسبان .

١١ - ٢١ كيف يعالج الاحتياطى عندما تجدد الملفات فى نفس المكان ؟

يمكن إعداد نسخة من الملف قبل إدخال العمليات الجارية . ويمكن حفظ هذه النسخة مع العمليات الجارية ، وتستخدم فى إعادة بناء الملف الرئيسى فى حالة حدوث تلف للملف الرئيسى .

١١ - ٢٢ تشكل المسألتان ٣٠ ، ٣١ من الفصل الثامن ما يشبه تسلسل تجديد ملف تتابعى ، بينما تكون المسألة ٣٠ برنامج

تنقيح ، وتؤدى المسألة ٣١ تجديد الملف . راجع المسألة ٣٠ لاستخدام جدول بأعداد الأيام فى كل شهر للتأكد من صحة حقل التاريخ فى بطاقات مدخلات الوقت . (اهمل الحالات الخاصة بنهاية القرن ، وبالسنوات الكبيسة) .

يعطى جزء من برنامج كويل فى شكل (١١ - ٥) ، وتعرف السطور من ٢٣٤ - ٢٥٦ ، جدول التاريخ ، تجرى السطور من ٢٦٩ - ٢٩٠ ، ومن ٢٩٩ - ٣١٠ التأكد من المدخلات .

(THE FOLLOWING TABLE IS IN WORKING-STORAGE) . . .

```

00110      01  WS-VALID-DATES-TABLE.
00111      05  VALID-DAYS-PER-MONTH-VALUES.
00112          10  FILLER                PIC 99          VALUE 31.
00113          10  FILLER                PIC 99          VALUE 28.
00114          10  FILLER                PIC 99          VALUE 31.
00115          10  FILLER                PIC 99          VALUE 30.
00116          10  FILLER                PIC 99          VALUE 31.
00117          10  FILLER                PIC 99          VALUE 30.
00118          10  FILLER                PIC 99          VALUE 31.
00119          10  FILLER                PIC 99          VALUE 31.
00120          10  FILLER                PIC 99          VALUE 30.
00121          10  FILLER                PIC 99          VALUE 31.
00122          10  FILLER                PIC 99          VALUE 30.
00123          10  FILLER                PIC 99          VALUE 31.
00124      05  DAYS-PER-MONTH      REDEFINES VALID-DAYS-PER-MONTH-VALUES
00125          PIC 99
00126          OCCURS 12 TIMES.

```

(THE FOLLOWING ARE THE RELEVANT PROCEDURE DIVISION PARAGRAPHS) . . .

```

00269      240-VALIDATE-FIELDS.
00270
00271          MOVE "NO"      TO WS-ERROR-DETECTED-SW
00272          MOVE SPACES TO WS-UNDER-LINE
00273          IF TIME-CARD-ID NOT NUMERIC
00274              MOVE ALL "-" TO WS-UNDER-ID
00275              MOVE "YES"   TO WS-ERROR-DETECTED-SW
00276
00277          IF TIME-CARD-HOURS NOT NUMERIC
00278              MOVE ALL "-" TO WS-UNDER-HOURS
00279              MOVE "YES"   TO WS-ERROR-DETECTED-SW
00280
00281          IF TIME-CARD-CLOSING-DATE NOT NUMERIC
00282              MOVE ALL "-" TO WS-UNDER-DATE
00283              MOVE "YES"   TO WS-ERROR-DETECTED-SW
00284          ELSE
00285              PERFORM 260-VALIDATE-DATE
00286
00287          IF TIME-CARD-DEPARTMENT NOT NUMERIC
00288              MOVE ALL "-" TO WS-UNDER-DEPARTMENT
00289              MOVE "YES"   TO WS-ERROR-DETECTED-SW
00290
00298
00299      260-VALIDATE-DATE.
00300
00301          IF TIME-CARD-CLOSING-YR GREATER THAN SYSTEM-YY
00302              MOVE ALL "-" TO WS-UNDER-YR
00303              MOVE "YES" TO WS-ERROR-DETECTED-SW
00304
00305          IF TIME-CARD-CLOSING-MO LESS THAN 1 OR GREATER THAN 12
00306              MOVE ALL "-" TO WS-UNDER-MO
00307              MOVE "YES" TO WS-ERROR-DETECTED-SW
00308          ELSE IF TIME-CARD-CLOSING-DAY LESS THAN 1 OR
00309              GREATER THAN DAYS-PER-MONTH (TIME-CARD-CLOSING-MO)
00310              MOVE ALL "-" TO WS-UNDER-DAY
00311              MOVE "YES" TO WS-ERROR-DETECTED-SW
00312

```


١١ - ٢٣ اكتب شفرة برنامج كويل كاملا للمسألة رقم ٣١ من الفصل الثامن . واستخدم فحص جدول الضرائب فى حساب ضريبة الدخل ، وجدد الملف الرئيسى للعاملين (فى موقعه) ، وذلك بإضافة إجمالى الأجر ، وقيمة الضرائب ، وصافى الأجر الى الحقول المناظرة لها من أول العام حتى الآن داخل السجل الرئيسى للعاملين .
انظر شكل ١١ - ٦ . توجد حسابات الضرائب فى السطور من ١١٢ - ١٣٧ ، ومن ٢٣١ - ٢٤٣ التجديد فى السطور من ٢٢٤ - ٢٥٦ ، ومن السطور ٢٠٩ - ٣١٥ .

```

00001 IDENTIFICATION DIVISION.
00002
00003 PROGRAM-ID. PAYROLL.
00004
00005 AUTHOR. LARRY NEWCOMER.
00006 INSTALLATION. PENN STATE UNIVERSITY -- YORK CAMPUS.
00007
00008 ENVIRONMENT DIVISION.
00009
00010 CONFIGURATION SECTION.
00011 SOURCE-COMPUTER. IBM-3081.
00012 OBJECT-COMPUTER. IBM-3081.
00013
00014 INPUT-OUTPUT SECTION.
00015 FILE-CONTROL.
00016
00017 SELECT VALID-TIME-FILE
00018 ASSIGN TO DISKTIME
00019 ORGANIZATION IS SEQUENTIAL
00020 ACCESS IS SEQUENTIAL
00021
00022 SELECT EMPLOYEE-MASTER-FILE
00023 ASSIGN TO EMPMAST
00024 ORGANIZATION IS SEQUENTIAL
00025 ACCESS IS SEQUENTIAL
00026
00027 SELECT PAYROLL-CHECK-FILE
00028 ASSIGN TO PAYCHECK
00029 ORGANIZATION IS SEQUENTIAL
00030 ACCESS IS SEQUENTIAL
00031
00032 SELECT PAYROLL-REGISTER-FILE
00033 ASSIGN TO REGISTER
00034 ORGANIZATION IS SEQUENTIAL
00035 ACCESS IS SEQUENTIAL
00036
00037
00038 DATA DIVISION.
00039
00040 FILE SECTION.
00041
00042 FD VALID-TIME-FILE
00043 BLOCK CONTAINS 0 RECORDS
00044 RECORD CONTAINS 80 CHARACTERS
00045 LABEL RECORDS ARE STANDARD
00046
00047
00048 01 VALID-TIME-RECORD.
00049 05 VALID-TIME-ID PIC X(5).
00050 05 VALID-TIME-HOURS PIC S9(2)V9.
00051 05 VALID-TIME-CLOSING-DATE PIC X(6).
00052 05 VALID-TIME-DEPARTMENT PIC X(4).
00053 05 VALID-TIME-EDIT-DATE PIC X(6).

```

شكل (١١ - ٦)

```

00054          05 FILLER PIC X(56).
00055
00056      FD EMPLOYEE-MASTER-FILE
00057          BLOCK CONTAINS 0 RECORDS
00058          RECORD CONTAINS 80 CHARACTERS
00059          LABEL RECORDS ARE STANDARD
00060
00061
00062      01 EMPLOYEE-MASTER-RECORD.
00063          05 EMPLOYEE-MASTER-ID PIC X(5).
00064          05 EMPLOYEE-MASTER-NAME PIC X(20).
00065          05 EMPLOYEE-MASTER-RATE PIC S9(3)V99.
00066          05 EMPLOYEE-MASTER-YTD-GROSS PIC 9(7)V99.
00067          05 EMPLOYEE-MASTER-YTD-TAX PIC 9(6)V99.
00068          05 EMPLOYEE-MASTER-YTD-NET PIC 9(7)V99.
00069          05 FILLER PIC X(24).
00070
00071
00072      FD PAYROLL-CHECK-FILE
00073          BLOCK CONTAINS 0 RECORDS
00074          RECORD CONTAINS 66 CHARACTERS
00075          LABEL RECORDS ARE STANDARD
00076
00077
00078      01 PAYROLL-CHECK-RECORD.
00079          05 PAYROLL-ID PIC X(5).
00080          05 PAYROLL-HOURS PIC S9(2)V9.
00081          05 PAYROLL-DATE PIC X(6).
00082          05 PAYROLL-DEPARTMENT PIC X(4).
00083          05 PAYROLL-RATE PIC S9(3)V99.
00084          05 PAYROLL-NAME PIC X(20).
00085          05 PAYROLL-GROSS PIC S9(6)V99.
00086          05 PAYROLL-TAX PIC S9(5)V99.
00087          05 PAYROLL-NET PIC S9(6)V99.
00088
00089
00089      FD PAYROLL-REGISTER-FILE
00090          RECORD CONTAINS 132 CHARACTERS
00091          LABEL RECORDS ARE OMITTED
00092
00093
00094      01 ERROR-REPORT-LINE PIC X(132).
00095
00096      WORKING-STORAGE SECTION.
00097
00098      01 PROGRAM-SWITCHES.
00099          05 WS-END-TIME-FILE-SW PIC X(3).
00100              88 NO-MORE-TIMES VALUE "YES".
00101              88 TIME-AVAILABLE VALUE "NO".
00102          05 WS-END-MASTER-FILE-SW PIC X(3).
00103              88 NO-MORE-MASTERS VALUE "YES".
00104              88 MASTER-AVAILABLE VALUE "NO".
00105
00106      01 PROGRAM-COUNTERS.
00107          05 WS-PAGE-NUMBER PIC S9(3) COMP-3.
00108          05 WS-NUMBER-EMPLOYEES PIC S9(5) COMP-3.
00109          05 WS-NUMBER-TO-SKIP PIC S9 COMP SYNC.
00110          05 WS-LINES-ON-PAGE PIC S9(2) COMP SYNC.
00111

```

```

00112 * -----
00113 *
00114 * TAX TABLE IS USED TO LOOKUP TAX-RATE. NOTE HOW ENTRIES
00115 * ARE ARRANGED IN TABLE. LAST ENTRY = ZERO GUARANTEES
00116 * SEARCH WILL ALWAYS SUCCEED. ALTHOUGH WE USE THE VALUE
00117 * AND REDEFINES CLAUSES TO LOAD THIS TABLE.
00118 * IT COULD ALSO BE INPUT FROM A FILE.
00119 * -----
00120 *
00121 *
00122 01 TAX-DATA.
00123 05 TAX-TABLE-VALUES.
00124 10 FILLER PIC S9(3)V99 COMP-3 VALUE +800.00.
00125 10 FILLER PIC SV999 COMP-3 VALUE +.25.
00126 10 FILLER PIC S9(3)V99 COMP-3 VALUE +500.00.
00127 10 FILLER PIC SV999 COMP-3 VALUE +.15.
00128 10 FILLER PIC S9(3)V99 COMP-3 VALUE +300.00.
00129 10 FILLER PIC SV999 COMP-3 VALUE +.08.
00130 10 FILLER PIC S9(3)V99 COMP-3 VALUE ZERO.
00131 10 FILLER PIC SV999 COMP-3 VALUE +.05.
00132 05 TAX-TABLE REDEFINES TAX-TABLE-VALUES
00133 OCCURS 4 TIMES
00134 INDEXED BY TAX-INDEX.
00135 10 LOWER-LIMIT PIC S9(3)V99 COMP-3.
00136 10 TAX-RATE PIC SV999 COMP-3.
00137
00138 01 PROGRAM-TOTAL-AREAS.
00139 05 WS-TOTAL-HOURS PIC S9(7)V9 COMP-3.
00140
00141 01 PROGRAM-COMPUTATION-AREAS.
00142 05 WS-GROSS-PAY PIC S9(7)V99 COMP-3.
00143 05 WS-NET-PAY PIC S9(7)V99 COMP-3.
00144 05 WS-TAX PIC S9(5)V99 COMP-3.
00145
00146 01 PROGRAM-CONSTANTS.
00147 05 WS-PAGE-SIZE PIC S9(2) COMP SYNC
00148 VALUE +50.
00149 05 WS-SKIP-BEFORE-HEADING PIC S9(2) COMP SYNC
00150 VALUE +2.
00151 05 WS-SKIP-BEFORE-FOOTING PIC S9(2) COMP SYNC
00152 VALUE +4.
00153 05 WS-SKIP-BEFORE-DETAIL PIC S9(2) COMP SYNC
00154 VALUE +2.
00155 05 WS-UNMATCHED-TIME-MESSAGE PIC X(30)
00156 VALUE "EMPLOYEE NOT ON FILE".
00157 05 WS-UNMATCHED-MASTER-MESSAGE PIC X(30)
00158 VALUE "INACTIVE THIS PAY PERIOD".
00159 05 WS-OVERTIME-POINT PIC S9V9 COMP-3
00160 VALUE +40.0.
00161 05 WS-OVERTIME-FACTOR PIC S9V9 COMP-3
00162 VALUE +1.5.
00163
00164 01 WS-SYSTEM-DATE.
00165 05 SYSTEM-YY PIC 99.
00166 05 SYSTEM-MM PIC 99.
00167 05 SYSTEM-DD PIC 99.
00168
00169

```

```

00170
00171      01 WS-HEADING-LINE-1.
00172      05 FILLER          PIC X(2)          VALUE SPACES.
00173      05 FILLER          PIC X(18)         VALUE "PAYROLL REGISTER".
00174
00175      05 WS-HEADING-MM    PIC Z9.
00176      05 FILLER          PIC X            VALUE "/".
00177      05 WS-HEADING-DD    PIC 99.
00178      05 FILLER          PIC X            VALUE "/".
00179      05 WS-HEADING-YY    PIC 99.
00180      05 FILLER          PIC X(3)         VALUE SPACES.
00181      05 FILLER          PIC X(5)         VALUE "PAGE ".
00182      05 WS-HEADING-PAGE  PIC ZZ9.
00183      05 FILLER          PIC X(93)        VALUE SPACES.
00184
00185      01 WS-HEADING-LINE-2.
00186      05 FILLER          PIC X(7)         VALUE " ID".
00187      05 FILLER          PIC X(5)         VALUE "HRS".
00188      05 FILLER          PIC X(8)         VALUE " DATE".
00189      05 FILLER          PIC X(6)         VALUE "DEPT".
00190      05 FILLER          PIC X(8)         VALUE " RATE".
00191      05 FILLER          PIC X(98)        VALUE "NET PAY".
00192
00193      01 WS-DETAIL-LINE.
00194      05 WS-DETAIL-ID     PIC X(5).
00195      05 FILLER          PIC X(1)         VALUE SPACES.
00196      05 WS-DETAIL-HOURS  PIC ZZ.9.
00197      05 FILLER          PIC X(2)         VALUE SPACES.
00198      05 WS-DETAIL-DATE   PIC X(6).
00199      05 FILLER          PIC X(2)         VALUE SPACES.
00200      05 WS-DETAIL-DEPARTMENT PIC X(4).
00201      05 FILLER          PIC X(2)         VALUE SPACES.
00202      05 WS-DETAIL-RATE   PIC ZZZ.99.
00203      05 FILLER          PIC X(2)         VALUE SPACES.
00204      05 WS-DETAIL-REMARKS.
00205      10 WS-DETAIL-PAY   PIC ZZZ,ZZZ.99.
00206      10 FILLER          PIC X(88).
00207
00208      01 WS-FOOTING-LINE.
00209      05 FILLER          PIC X(12)        VALUE "# EMPLOYEES-".
00210
00211      05 WS-FOOTING-COUNT  PIC ZZ,ZZ9.
00212      05 FILLER          PIC X(3)         VALUE SPACES.
00213      05 FILLER          PIC X(12)        VALUE "TOTAL HOURS-".
00214
00215      05 WS-FOOTING-TOTAL  PIC Z,ZZZ,ZZZ.9.
00216      05 FILLER          PIC X(88)        VALUE SPACES.
00217
00218      01 WS-OUTPUT-AREA   PIC X(132).
00219
00220      PROCEDURE DIVISION.
00221
00222      CREATE-PAYROLL-CHECK-FILE.
00223
00224          PERFORM 100-INITIALIZE
00225          PERFORM 200-MATCH-TIME-CARDS-TO-MASTER
00226              UNTIL NO-MORE-TIMES AND NO-MORE-MASTERS
00227          PERFORM 300-TERMINATE
00228          STOP RUN
00229
00230

```

شكل (١١ - ٦) تكملة

```

00231      100-INITIALIZE.
00232
00233      OPEN      INPUT      VALID-TIME-FILE
00234                  I-O      EMPLOYEE-MASTER-FILE
00235                  OUTPUT    PAYROLL-REGISTER-FILE
00236                  PAYROLL-CHECK-FILE
00237      MOVE "NO" TO      WS-END-TIME-FILE-SW
00238                  WS-END-MASTER-FILE-SW
00239      MOVE ZERO TO      WS-PAGE-NUMBER
00240                  WS-NUMBER-EMPLOYEES
00241                  WS-TOTAL-HOURS
00242      MOVE WS-PAGE-SIZE TO      WS-LINES-ON-PAGE
00243      ACCEPT WS-SYSTEM-DATE FROM DATE
00244      MOVE SYSTEM-YY TO      WS-HEADING-YY
00245      MOVE SYSTEM-MM TO      WS-HEADING-MM
00246      MOVE SYSTEM-DD TO      WS-HEADING-DD
00247
00248      PERFORM 270-GET-TIME-RECORD
00249      PERFORM 280-GET-MASTER-RECORD
00250
00251      B-MATCH-TIME-CARDS-TO-MASTER.
00252
00253      MOVE SPACES TO WS-DETAIL-LINE
00254      IF VALID-TIME-ID EQUAL EMPLOYEE-MASTER-ID
00255          PERFORM 210-CREATE-PAYROLL-REC
00256          PERFORM 215-UPDATE-MASTER-FILE
00257          PERFORM 270-GET-TIME-RECORD
00258          PERFORM 280-GET-MASTER-RECORD
00259      ELSE IF VALID-TIME-ID LESS THAN EMPLOYEE-MASTER-ID
00260          MOVE WS-UNMATCHED-TIME-MESSAGE
00261                  TO WS-DETAIL-REMARKS
00262          MOVE VALID-TIME-ID      TO WS-DETAIL-ID
00263          MOVE VALID-TIME-HOURS   TO WS-DETAIL-HOURS
00264          MOVE VALID-TIME-CLOSING-DATE
00265                  TO WS-DETAIL-DATE
00266          MOVE VALID-TIME-DEPARTMENT
00267                  TO WS-DETAIL-DEPARTMENT
00268          MOVE WS-SKIP-BEFORE-DETAIL
00269                  TO WS-NUMBER-TO-SKIP
00270          MOVE WS-DETAIL-LINE     TO WS-OUTPUT-AREA
00271          PERFORM 270-GET-TIME-RECORD
00272      ELSE
00273          MOVE WS-UNMATCHED-MASTER-MESSAGE
00274                  TO WS-DETAIL-REMARKS
00275          MOVE EMPLOYEE-MASTER-ID TO WS-DETAIL-ID
00276          MOVE EMPLOYEE-MASTER-RATE TO WS-DETAIL-RATE
00277          MOVE WS-SKIP-BEFORE-DETAIL TO WS-NUMBER-TO-SKIP
00278          MOVE WS-DETAIL-LINE     TO WS-OUTPUT-AREA
00279          PERFORM 280-GET-MASTER-RECORD
00280
00281      PERFORM 260-PRINT-REGISTER
00282
00283      210-CREATE-PAYROLL-REC.
00284
00285
00286      MOVE VALID-TIME-ID      TO WS-DETAIL-ID
00287      MOVE VALID-TIME-HOURS   TO WS-DETAIL-HOURS
00288      MOVE VALID-TIME-CLOSING-DATE TO WS-DETAIL-DATE
00289      MOVE VALID-TIME-DEPARTMENT TO WS-DETAIL-DEPARTMENT
00290      MOVE EMPLOYEE-MASTER-RATE TO WS-DETAIL-RATE

```

```

00291      MOVE WS-SKIP-BEFORE-DETAIL      TO WS-NUMBER-TO-SKIP
00292
00293      PERFORM 220-CALCULATE-PAY
00294      MOVE WS-NET-PAY                    TO WS-DETAIL-PAY
00295      PERFORM 250-WRITE-PAYROLL-RECORD
00296      MOVE WS-DETAIL-LINE TO WS-OUTPUT-AREA
00297      ADD 1                                TO WS-NUMBER-EMPLOYEES
00298      ADD VALID-TIME-HOURS                TO WS-TOTAL-HOURS
00299
00300
00301      220-CALCULATE-PAY.
00302
00303      PERFORM 230-CALCULATE-GROSS
00304      PERFORM 240-CALCULATE-TAX
00305      SUBTRACT WS-TAX FROM WS-GROSS-PAY
00306      GIVING WS-NET-PAY
00307
00308
00309      215-UPDATE-MASTER-FILE.
00310
00311      ADD WS-TAX      TO EMPLOYEE-MASTER-YTD-TAX
00312      ADD WS-GROSS-PAY TO EMPLOYEE-MASTER-YTD-GROSS
00313      ADD WS-NET-PAY  TO EMPLOYEE-MASTER-YTD-NET
00314      REWRITE EMPLOYEE-MASTER-RECORD
00315
00316
00317      230-CALCULATE-GROSS.
00318
00319      IF VALID-TIME-HOURS GREATER THAN WS-OVERTIME-POINT
00320      COMPUTE WS-GROSS-PAY =
00321          (WS-OVERTIME-POINT * EMPLOYEE-MASTER-RATE)
00322      +   ((VALID-TIME-HOURS - WS-OVERTIME-POINT)
00323          * EMPLOYEE-MASTER-RATE * WS-OVERTIME-FACTOR)
00324      ELSE
00325      COMPUTE WS-GROSS-PAY =
00326          VALID-TIME-HOURS * EMPLOYEE-MASTER-RATE
00327
00328
00329      240-CALCULATE-TAX
00330
00331      * -----
00332      *
00333      * USE SEQUENTIAL SEARCH TO LOOKUP TAX RATE IN TAX TABLE
00334      * (DUE TO DESIGN OF TABLE, AT END... CONDITION CAN'T OCCUR)
00335      *
00336      * -----
00337
00338      SET TAX-INDEX TO 1
00339      SEARCH TAX-TABLE
00340      WHEN WS-GROSS-PAY GREATER THAN LOWER-LIMIT (TAX-INDEX)
00341      COMPUTE WS-TAX =
00342          WS-GROSS-PAY * TAX-RATE (TAX-INDEX)
00343
00344
00345      250-WRITE-PAYROLL-RECORD.
00346
00347      MOVE VALID-TIME-ID      TO PAYROLL-ID
00348      MOVE VALID-TIME-HOURS   TO PAYROLL-HOURS
00349      MOVE VALID-TIME-CLOSING-DATE TO PAYROLL-DATE
00350      MOVE VALID-TIME-DEPARTMENT TO PAYROLL-DEPARTMENT

```

```

00351      MOVE EMPLOYEE-MASTER-RATE      TO PAYROLL-RATE
00352      MOVE EMPLOYEE-MASTER-NAME      TO PAYROLL-NAME
00353      MOVE WS-GROSS-PAY              TO PAYROLL-GROSS
00354      MOVE WS-TAX                    TO PAYROLL-TAX
00355      MOVE WS-NET-PAY                TO PAYROLL-NET
00356      WRITE PAYROLL-CHECK-RECORD
00357      .
00358
00359      260-PRINT-REGISTER.
00360
00361      IF WS-LINES-ON-PAGE + WS-NUMBER-TO-SKIP
00362      GREATER THAN WS-PAGE-SIZE
00363          ADD 1 TO WS-PAGE-NUMBER
00364          MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE
00365          WRITE ERROR-REPORT-LINE
00366              FROM WS-HEADING-LINE-1
00367              AFTER ADVANCING PAGE
00368          WRITE ERROR-REPORT-LINE
00369              FROM WS-HEADING-LINE-2
00370              AFTER ADVANCING WS-SKIP-BEFORE-HEADING LINES
00371          MOVE WS-SKIP-BEFORE-HEADING TO WS-LINES-ON-PAGE
00372      .
00373      WRITE ERROR-REPORT-LINE
00374          FROM WS-OUTPUT-AREA
00375          AFTER ADVANCING WS-NUMBER-TO-SKIP LINES
00376      ADD WS-NUMBER-TO-SKIP TO WS-LINES-ON-PAGE
00377      .
00378
00379      270-GET-TIME-RECORD.
00380
00381      READ VALID-TIME-FILE
00382      AT END
00383          MOVE "YES" TO WS-END-TIME-FILE-SW
00384          MOVE HIGH-VALUES TO VALID-TIME-ID
00385
00386
00387      280-GET-MASTER-RECORD.
00388
00389      READ EMPLOYEE-MASTER-FILE
00390      AT END
00391          MOVE "YES" TO WS-END-MASTER-FILE-SW
00392          MOVE HIGH-VALUES TO EMPLOYEE-MASTER-ID
00393      .
00394
00395      300-TERMINATE.
00396
00397      MOVE WS-NUMBER-EMPLOYEES      TO WS-FOOTING-COUNT
00398      MOVE WS-TOTAL-HOURS          TO WS-FOOTING-TOTAL
00399      MOVE WS-FOOTING-LINE          TO WS-OUTPUT-AREA
00400      MOVE WS-SKIP-BEFORE-FOOTING TO WS-NUMBER-TO-SKIP
00401      MOVE WS-FOOTING-LINE          TO WS-OUTPUT-AREA
00402      PERFORM 260-PRINT-REGISTER
00403
00404      CLOSE      VALID-TIME-FILE
00405                EMPLOYEE-MASTER-FILE
00406                PAYROLL-CHECK-FILE
00407                PAYROLL-REGISTER-FILE
00408

```


الفصل الثانى عشر

ترتيب ودمج الملفات

File Sorting and Merging

يشمل ترتيب ودمج الملفات إعادة ترتيب السجلات المنطقية فى الملف ، والتي تكون موجودة فى وحدة تخزين مساعد . وهذا يضاهى ترتيب الجدول ، الذى يحدث فيه إعادة ترتيب لأجزاء من سجل منطقى (أى الجدول) . ويعد ترتيب الملفات سهلاً فى برمجته فى الكويل باستخدام فعل الترتيب SORT .

١٢ - ١ معجم ترتيب الملفات

تسمى الحقول الموجودة فى سجل منطقى ، والتي يرتب على أساسها الملف ، حقول تحكم control fields ، أو مفاتيح الترتيب sort keys . يمكن ترتيب السجلات فى ملف فى تتابع تصاعدى ascending ، أو تتابع تنازلى descending طبقاً لـى حقل . يسمح كويل IBM OS/VS بالترتيب الآتى للملف ، طبقاً لعدد يصل إلى 12 حقل مختلف ، ويأتى خليط من التتابع التصاعدى أو التنازلى . عندما يؤدى الترتيب على مفاتيح متعددة ... يسمى الحقل الأكثر أهمية بالمفتاح الرئيسى major key ويسمى الحقل الأقل أهمية بالمفتاح الأدنى minor key ، أما بقية الحقول - فتسمى بالمفاتيح المتوسطة intermediate keys - والمفتاح الأدنى كذلك - فهو يؤثر على الترتيب داخل كل قيمة من قيم المفتاح الرئيسى . وإذا ركزنا بالتالى الانتباه على حقل متوسط أو حقل أدنى فقط .. فلن يبدو الملف ككل بأنه مرتب .

مثال ١٢ - ١ :

يعرض جدول (١٢ - ١) ترتيباً للملف مبيعات طبقاً لثلاثة مفاتيح . الترتيب الرئيسى تصاعدى ، والترتيب المتوسط - داخل الترتيب الرئيسى - تنازلى ، والترتيب الأدنى - داخل الترتيب المتوسط - تصاعدى .

جدول (١٢ - ١)

رقم القسم (مفتاح رئيسى)	رقم البائع (مفتاح متوسط)	رقم العميل (مفتاح أدنى)	قيمة المبيعات (ليس حقل مفتاح)
101	7823	10925	1,000.98
101	7823	27832	2,000.00
101	7823	59838	1,500.00
101	6345	20782	1,200.32
101	6345	48793	5,000.00
101	3287	10925	3,003.43
206	8291	27832	1,305.50
206	8291	40058	2,783.45
206	5094	39840	5,672.34
206	5094	60291	3,476.52
206	4925	27832	4,000.00
206	4925	30618	2,987.50
206	3920	10071	4,526.98

١٢ - ٢ استخدام عبارة الترتيب

عبارة الترتيب - الموضح تكوينها فى شكل (١٢ - ١) تكتب فى جزء الإجراءات ، ومعها يجب استخدام ملف ترتيب sort file افتراضى ، ويجب أن تكون لهذا الملف عبارة SELECT فى جزء الأوساط ، وجزء وصف ملف SD يشبه وصف الملف FD فى جزء البيانات . ويعرف وصف السجل ملف الترتيب الحقول الرئيسية (حقول المفاتيح) التى يؤدى الترتيب طبقا لها . وتستخدم منطقة السجل المنطقى ملف الترتيب فى تمرير سجلات من وإلى برنامج الكويل ، وبرنامج منفعة سابق الإعداد للترتيب والدمج sort merge utility program ، والذى يجرى الترتيب الفعلى . وتأتى برامج المنفعة مع نظام التشغيل .

SORT file-name-1

ON { ASCENDING } KEY data-name-1 [data-name-2] ...

[ON { ASCENDING } KEY data-name-3 [data-name-4] ...] ...

[COLLATING SEQUENCE IS alphabet-name]

{ USING file-name-2 [file-name-3] ...

{ INPUT PROCEDURE IS section-name-1 [{ THROUGH } section-name-2]

{ GIVING file-name-4

{ OUTPUT PROCEDURE IS section-name-3 [{ THROUGH } section-name-4]

شكل (١٢ - ١)

.....
 ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.

 SELECT MORTGAGE-PAYMENT-FILE
 ASSIGN TO MORTRECV
 ORGANIZATION IS SEQUENTIAL
 ACCESS IS SEQUENTIAL

 SELECT PAYMENT-SORT-FILE
 ASSIGN TO SORTWK

.....
 DATA DIVISION.
 FILE SECTION.

FD MORTGAGE-PAYMENT-FILE
 BLOCK CONTAINS 0 CHARACTERS
 RECORD CONTAINS 23 CHARACTERS
 LABEL RECORDS ARE STANDARD

01 MORTGAGE-PAYMENT-RECORD.
 05 MORTGAGE-ACCOUNT-ID PIC X(6).
 05 MORTGAGE-DUE-DATE PIC X(6).
 05 MORTGAGE-PAID-DATE PIC X(6).
 05 MORTGAGE-AMOUNT PIC S9(3)V99.

SD PAYMENTS-SORT-FILE
 RECORD CONTAINS 23 CHARACTERS

01 SORT-RECORD.
 05 SORT-ACCOUNT-ID PIC X(6).
 05 SORT-DUE-DATE PIC X(6).
 05 SORT-PAID-DATE PIC X(6).
 05 SORT-AMOUNT PIC S9(3)V99.

.....
 PROCEDURE DIVISION.

.....
 SORT PAYMENTS-SORT-FILE
 ON ASCENDING KEY SORT-ACCOUNT-ID
 ON ASCENDING KEY SORT-PAID-DATE
 INPUT PROCEDURE IS SCREEN-PAYMENT-RECORDS
 OUTPUT PROCEDURE IS PRODUCE-QUARTERLY-REPORT

.....
 SCREEN-PAYMENT-RECORDS SECTION.

.....
 PRODUCE-QUARTERLY-REPORT SECTION.

هنا .. يعتبر الملف MORTAGE - PAYMENT - FILE هو الملف الفعلى على القرص ، والمراد ترتيبه تصاعديا طبقا لتاريخ الدفع (المفتاح الأدنى) ، داخل رقم تعريف الحساب (المفتاح الرئيسى) ، وملف الترتيب هو - SORT - PAYMENTS - FILE . لاحظ ما يلى :

(١) بالنسبة لملف الترتيب .. تأخذ عبارة SELECT الشكل المبسط التالى :

SELECT sort-file-name
ASSIGN TO assignment-name-comment

ونظراً لأن ملف الترتيب ليس ملفاً فعلياً .. يعامل اسم ASSIGN TO كتعليق . ولا تكون هناك حاجة إلى لغة تحكم العمل إضافية : لتعريف ملف الترتيب .

(٢) عادة ما تكون عبارات تحكم العمل أو عبارات لغة الأوامر مطلوبة إذا كانت عبارة SORT تستدعى منفعة الترتيب أو الدمج . وعلى القارئ أن يحدد المطلوب بالنسبة للجهاز المتاح له استخدامه .

(٣) الملف FILE - SORT - PAYMENT معرف فى جزء البيانات بواسطة SD :

SD sort-file-name
[RECORD CONTAINS [integer-1 TO] integer-2 CHARACTERS]
[DATA {RECORD IS
RECORDS ARE} data-name]

(٤) يقدم SORT - RECORD (المستوى 01) وصفاً للسجلات المراد ترتيبها ، ويعمل كذاكرة احتياطية ، تمرر فيها السجلات من وإلى منفعة الترتيب إلى ومن البرنامج .

(٥) تسمى عبارة SORT ملف الترتيب ، وليس الملف الفعلى . ويعرف جزء ASCENDING KEY أو DESCENDING KEY حقول مفاتيح الترتيب من الحقل الرئيسى إلى الحقل الأدنى ، كما يجب أن تظهر عناصر البيانات هذه فى وصف سجل ملف الترتيب ، الذى يبدأ بتنفيذ INPUT PROCEDURE .

(٦) يجب أن يكون INPUT PROCEDURE قسماً . ويحتوى SCREEN - PAYMENT - RECORDS على دورة برنامج تقرأ كل سجل منطلق من MORTAGE - PAYMENT - FILE ، وتنقله إلى SORT - RECORD ، وتسلمه إلى منفعة الترتيب بتنفيذ عبارة RELEASE . وعندما تنفذ عبارة SORT .. فإن INPUT PROCEDURE ينفذ مرة واحدة .

(٧) عندما ينتهى تنفيذ INPUT PROCEDURE .. ترتب منفعة الترتيب - بصورة تلقائية - السجلات المنطقية التى أزيحت إليه . وعند إتمام الترتيب ينفذ OUTPUT PROCEDURE (PRODUCE - QUARTERLY - REPORT) مرة واحدة .

(٨) يجب أن يكون OUTPUT PROCEDURE قسماً كذلك ، ويكتب ليطلب إعادة منفعة الترتيب (إلى SORT - RECORD) السجل المنطقى التالى فى تتابع الترتيب . ويحدث هذا بتنفيذ عبارة RETURN داخل OUTPUT PROCEDURE . عند ذلك يستطيع .. OUTPUT PROCEDURE تشغيل السجل (وفى هذه الحالة .. فهو يطبع سطرًا من سطور التقرير) .

كما فى INPUT PROCEDURE .. فإن PRODUCE - QUARTERLY - REPORT يحتوى على بكرة: حيث يعاد RETURN الملف المرتب من منفعة الترتيب ، سجلا سجلا .

(٩) عندما ينتهى تنفيذ OUTPUT PROCEDURE .. يستمر تنفيذ البرنامج بالعبارة التى تلى عبارة SORT مباشرة . حيث إن INPUT PROCEDURE يزيج RELEASE سجلات منطقية إلى منفعة الترتيب سجلا سجلا : فيمكن استخدامه فى غريلة screen السجلات ، أو عمل تشغيل مسبق preprocess قبل ترتيبها ، وفى الواقع ، إذا لم تكن هناك حاجة إلى الغريلة أو التشغيل المسبق للسجلات المراد ترتيبها .. يمكن حذف INPUT PROCEDURE ، وتستطيع منفعة الترتيب الحصول على السجلات المراد ترتيبها من الملف الفعلى مباشرة ، ويحدد هذا البديل بجزء USING فى عبارة SORT .

مثال ١٢ - ٣ :

افرض أنه مطلوب طباعة عناوين بريدية من الملف الرئيسى للعملاء : مرتبة طبقا لرقم العميل ، وأنه من المرغوب فيه إرسال خطابات إلى العملاء الذين لم يقوموا بعمليات شراء خلال آخر ٨ شهور فقط . ولكي تقل تكاليف البريد .. يجب أن ترتب العناوين طبقا للرقم البريدي : حيث يفترض أن معظم العملاء كانوا نشطين خلال آخر ٨ شهور ، لذا فمن الممكن استخدام INPUT PROCEDURE لإزاحة RELEASE السجلات الخاصة بالعملاء ، المراد طباعة عناوينهم فقط إلى منفعة الترتيب . أكثر من هذا .. يمكن تعريف سجلات ملف الترتيب بأنها تحتوى على حقول الاسم والعنوان فقط من سجلات الملف الرئيسى للعملاء ، ويمكن كتابة INPUT PROCEDURE : لنقل هذه الحقول - فقط - إلى منطقة سجلات ملف الترتيب ، قبل إزاحتها RELEASE إلى منفعة الترتيب .

يمكن للترتيب أن يأخذ بعد ذلك وقتا أقل من الكمبيوتر ، حيث يجرى الترتيب على سجلات أصغر .

مثال ١٢ - ٤ :

افرض ان الملف الرئيسى للعملاء يحتوى على إجمالى مشتريات آخر سنة ، وإجمالى مشتريات السنة الحالية . ومن المرغوب فيه طباعة قائمة بالعملاء فى ترتيب تنازلى طبقا لنسبة التغيير السنوية من العام الماضى إلى العام الحالى . لعمل ذلك .. يجب وصف حقل خاص بنسبة التغيير السنوية ، كجزء من وصف سجل ملف الترتيب . ويمكن استخدام INPUT PROCEDURE فى إدخال سجلات الملف الرئيسى للعملاء : لحساب نسب التغيير السنوية ، ثم تنقل بعد ذلك هذه النسبة مع المعلومات المطلوبة من سجل الملف الرئيسى للعملاء إلى سجل الترتيب : لإزاحتها RELEASE إلى منطقة الترتيب .

نركز على أنه عند استخدام INPUT PROCEDURE ، لا تحتاج السجلات المراد ترتيبها أن تناظر السجلات الموجودة فى أى ملف فعلى (مثل : لا يوجد حقل بنسبة التغيير فى سجلات الملف الرئيسى للعملاء) . ومن ناحية أخرى .. فإنه عند استخدام USING .. يجب أن يعكس SD كلاً من : USAGE , PICTURE ، وهياكل السجلات فى ملف USING بالضبط

وكما أن INPUT PROCEDURE تمكن من إجراء تشغيل سابق للسجلات غير المرتبة فإن OUTPUT PROCEDURE يجرى تشغيلاً لاحقاً على السجلات المرتبة : فإذا لم تكن هناك حاجة للتشغيل اللاحق .. فمن الممكن استخدام جزء GIVING فى توجيه منفعة الترتيب ، إلى ونجس السجلات المرتبة فى ملف GIVING مباشرة .

مثال ١٢ - ٥ :

إذا استخدم الملف MORTGAGE - PAYMENT - FILE من مثال (١٢ - ٢) فى ملف تجديد الملف الرئيسى للقروض .. فيمكن أن يحتوى برنامج التنقيح الذى يختبر الصحة ويرتبها على مايلى :

```

SORT PAYMENTS-SORT-FILE
ON ASCENDING KEY SORT-ACCOUNT-ID
INPUT PROCEDURE IS VALIDATE-PAYMENT-RECORDS
GIVING VALID-TRANS-FILE

```

حيث VALID - TRANS - FILE هو ملف فعلى ، له عبارة SELECT ، FD معتادان .
 ويجب ذكر أن برنامج الكويل يمكن أن يحتوى على أى عدد من عبارات SORT (أو MERGE) ، وذلك بافتراض أنه لا يوجد تداخل بين العبارات فقط .
 بعد حصر العمل الشامل لعبارة SORT .. فإننا نوجه بقية هذا القسم إلى إلقاء نظرة فاحصة على أجزائها المنفصلة وصيغها المختلفة .

جزء المفتاح تصاعدي أو تنازلي

الغرض من جزء المفتاح تصاعدي أو تنازلي ASCENDING / DESCENDING KEY ، هو تعريف مواقع المفاتيح داخل السجلات المراد ترتيبها . وبالنسبة للسجلات متغيرة الطول .. فمن المطلوب أن كل حقل مفتاح يتواجد في نفس الموقع تماما داخل كل السجلات .

مثال ١٢ - ٦ :

اعتبر الملف التالي :

```
SD PARTS-SORT-FILE
RECORD CONTAINS 12 TO 22 CHARACTERS.

01 MADE-PART-RECORD.
05 MADE-PART-ID          PIC X(6).
05 QUANTITY-ON-HAND      PIC S9(5) COMP-3.
05 MADE-WAREHOUSE-CODE   PIC X(3).
01 BOUGHT-PART-RECORD.
05 BOUGHT-PART-ID        PIC X(6).
05 BOUGHT-QUANTITY-ON-HAND PIC S9(5) COMP-3.
05 BOUGHT-WAREHOUSE-CODE PIC X(3).
05 SUPPLIER-ID           PIC X(7).
05 SUPPLIER-PRICE        PIC S9(3)V99 COMP-3.
```

طول السجل هو 12 بالنسبة إلى MADE - PART - RECORD ، و 22 بالنسبة إلى BOUGHT - PART - RECORD ، ولكته - طبقا لما هو مطلوب - يحتل PART - ID نفس الموضع في كل من السجلين (البايت من ١ - ٦) ، وكذلك هو الحال بالنسبة للحقل WAERHOUSE - CODE : حيث يحتل الموضع من البايت ١٠ - ١٢ ، وعلى هذا .. يمكن أن تكتب عبارة SORT لهذا الملف على النحو التالي:

```
SORT PARTS-SORT-FILE
ASCENDING KEY MADE-PART-ID
DESCENDING KEY MADE-WAREHOUSE-CODE
INPUT PROCEDURE IS SCREEN-PARTS
OUTPUT PROCEDURE IS PRINT-REPORT
```

كما يمكن أن تأخذ الشكل التالى أيضا:

```

SORT PARTS-SORT-FILE
  ASCENDING KEY BOUGHT-PART-ID
  DESCENDING KEY BOUGHT-WAREHOUSE-CODE
  INPUT PROCEDURE IS SCREEN-PARTS
  OUTPUT PROCEDURE IS PRINT-REPORT

```

جزء تسلسل التتابع

يتماثل جزء تسلسل التتابع collating sequence مع جزء PROGRAM COLLATING SEQUENCE، الذى سبق شرحه فى القسم الثالث من الفصل الرابع، إلا أن تسلسل التتابع المحدد لعبارة SORT، لا يحتاج أن يكون هو نفسه مثل تسلسل التتابع المحدد لمقارنة عناصر حرفية عديدة فى جزء الإجراءات، ويجب أن يعرف alphabet-name (شكل ١٢ - ١) فى مقطع الأسماء الخاصة (القسم الرابع الفصل الرابع). عادة يحذف تسلسل التتابع من عبارة SORT، مع اعتبار تسلسل التتابع التقليدى هو المستخدم.

إجراء المدخلات والإزاحة، ...

إجراء المخرجات والإعادة ...

عند استخدام إجراء المخلات INPUT PROCEDURE، يجب أن يحتوى على عبارة إزاحة RELEASE واحدة على الأقل.

RELEASE record-name [FROM data-name]

عبارة الإزاحة تشبه عبارة الكتابة WRITE، ويمكن التفكير فيها فى واقع الأمر بأنها كتابة WRITE سجل ملف ترتيب فى منفعة الترتيب؛ حيث تجمع كل هذه السجلات، لترتيبها عندما ينتهى جزء إجراء المخلات INPUT PROCEDURE، ويعمل جزء FROM مثل جزء FROM المستخدم مع عبارة WRITE تماماً.

مثال ١٢ - ٧ :

فى مثال (١٢ - ٦) .. يمكن أن تشمل شفرة SCREEN - PARTS ما يلى:

```

IF MADE-PART
  RELEASE MADE-PART-RECORD
  FROM INPUT-RECORD
ELSE
  MOVE INPUT-ID TO BOUGHT-PART-ID
  MOVE INPUT-QUANTITY TO BOUGHT-QUANTITY-ON-HAND
  MOVE INPUT-CODE TO BOUGHT-WAREHOUSE-CODE
  MOVE INPUT-SUPPLIER TO SUPPLIER-ID
  MOVE INPUT-PRICE TO SUPPLIER-PRICE
  RELEASE BOUGHT-PART-RECORD

```

يجب أن يحتوى كل إجراء مخرجات OUTPUT PROCEDURE على عبارة إعادة RETURN واحدة على الأقل، التي تطلب - من منفعة الترتيب - وضع السجل المنطقي التالي بعد الترتيب في منطقة سجل ملف الترتيب :

RETURN sort-file-name RECORD [INTO data-name]
AT END imperative-statement

تشبه عبارة RETURN عبارة READ ، وتعمل RETURN... INTO مثل عمل READ ... INTO . أكثر من هذا .. يعمل جزء AT END (أى بعد انتهاء ملف الترتيب) مثل جزء AT END فى عبارة READ تماما .

رتب ... معطيا ... مستخدما

عندما لا تكون هناك حاجة إلى تشغيل للسجلات المنطقية ، يسبق أو يتبع الترتيب .. استخدم رتب ... معطيا ... مستخدما . SORT... GIVING... USING

إلا أن الكفاءة تزداد كثيرا إذا ما نفذت منفعة الترتيب كبرنامج مستقل stand- alone program (بدلا من استدعائها من الكويل). فى عبارة SORT من الكويل، يجب إغلاق ملفات GIVING,USING عندما تنفذ عبارة SORT.

رتب .. إجراء مدخلات ... معطيا ...

عند استخدام إجراء المدخلات INPUT PROCEDURE ، يجب تشكيل جزء الإجراءات كما فى مثال ١٢ . ٨ .

مثال ١٢ - ٨ :

PROCEDURE DIVISION.

SORT...

ON...

INPUT PROCEDURE IS SCREEN-INPUT

OUTPUT PROCEDURE IS PRINT-REPORT

STOP RUN

SCREEN-INPUT SECTION.

PRINT-REPORT SECTION.

END-SORT-PROCEDURES SECTION.

A-PARAGRAPH.

B-PARAGRAPH.

لاحظ أن فعل SORT وضع في الجزء المنفذ ، الذي يليه مباشرة STOP RUN . ويعطى استخدام STOP RUN أثناء تنفيذ INPUT / OUTPUT PROCEDURE نتائج غير متوقعة ، وينتهي البرنامج نهاية غير طبيعية في كويل IBM OS / VS

لاحظ كذلك أن END - SORT - PROCEDURES SECTION ، بدون عنوان ، ينتمي A- PAR- B- PARAGRAPH ، إلى PRINT - REPORT SECTION ؛ فإذا كان البرنامج ينفذ مثلاً SCREEN - INPUT ، أو PRINT - REPORT أيًا من المقطعين الخارجيين (كما هو مسموح به في كويل IBM OS/VS) ، وعلى ذلك فالاختيار الأفضل للعنوان يمكن أن يكون : PERFORMED - PARAGRAPHS SECTION ، أو ما يشبهه. انظر المسألة ٢٢ من هذا الفصل للمزيد عن استخدام PERFORM في INPUT / OUTPUT PROCEDURE .

ويمكن أن يكون أحد التطبيقات التقليدية لعبارة رتب ... إجراء مدخلات ... معطيا SORT... INPUT PROCEDURE... GIVING في برنامج تنقيح.

مثال ١٢ - ٩ :

نقح برنامج التنقيح الموجود في المسألة ٢٢ - الفصل الحادي عشر ، والمسألة ٣٠ - الفصل الثامن بإسخال عبارة SORT .. تستخدم INPUT PROCEDURE في غريلة السجلات غير الصحيحة ، وتستخدم جزء GIVING في وضع العمليات الجارية الصحيحة والمرتبطة في ملف قرص.

انظر شكل (١٢ - ٢) مع تركيز الانتباه إلى هيكل جزء الإجراءات. جزء الإجراءات الأصلي (والمعطى جزء فقط منه في شكل (١١ - ٥) أصبح إجراء مدخلات INPUT PROCEDURE لعبارة SORT . عبارة WRITE الأصلية (غير موجودة في شكل ١١ - ٥ ، لكن انظر السطر ٢٠٩ في شكل ٨ - ١٤ ، والتي حولت العمليات الجارية الصحيحة إلى القرص) استبدلت بعبارة RELEASE والتي تعطي العمليات الجارية الصحيحة لمنفعة الترتيب ، والتي تنتج FILE - TIME - VALID كاستجابة لجزء GIVING . لاحظ أن ملف GIVING يجب أن يفلق ، عندما تنفذ SORT ، حيث تفتح منفعة الترتيب OPEN ، وتغلق CLOSE ملفات GIVING ، USING GIVING .

```

00014      INPUT-OUTPUT SECTION.
00015      FILE-CONTROL.
00016
00017      SELECT TIME-CARD-FILE
00018      ASSIGN TO TIMECARD
00019      ORGANIZATION IS SEQUENTIAL
00020      ACCESS IS SEQUENTIAL
00021
00022      *
00023      *
00024      *
00025      * TO DO A COBOL SORT, ONE MUST DEFINE A SELECT STATEMENT
00026      * FOR THE "SORT FILE", WHICH IS NOT ACTUALLY A FILE LIKE
00027      * OTHER DATA FILES IN THE PROGRAM, BUT RATHER REPRESENTS
00028      * THE AREA IN WHICH THE SORT/MERGE UTILITY WILL REARRANGE
00029      * THE RECORDS TO BE SORTED.
00030      *
00031      *
00032      *

```

شكل (١٢ - ٢)

```

00033      SELECT SORTING-TIME-CARD-FILE
00034          ASSIGN TO SORTWK
00035
00036      SELECT VALID-TIME-FILE
00037          ASSIGN TO DISKTIME
00038          ORGANIZATION IS SEQUENTIAL
00039          ACCESS IS SEQUENTIAL
00040
00041      SELECT ERROR-LISTING-FILE
00042          ASSIGN TO ERRORLOG
00043          ORGANIZATION IS SEQUENTIAL
00044          ACCESS IS SEQUENTIAL
00045
00046
00047      DATA DIVISION.
00048
00049      FILE SECTION.
00050
00051      FD  TIME-CARD-FILE
00052          RECORD CONTAINS 80 CHARACTERS
00053          LABEL RECORDS ARE OMITTED
00054
00055
00056      01  TIME-CARD-RECORD.
00057          05  TIME-CARD-ID                PIC X(5).
00058          05  TIME-CARD-HOURS             PIC 9(2)V9.
00059          05  TIME-CARD-X-HOURS           REDEFINES TIME-CARD-HOURS
00060                                           PIC X(3).
00061          05  TIME-CARD-CLOSING-DATE.
00062              10  TIME-CARD-CLOSING-MO          PIC 99.
00063              10  TIME-CARD-CLOSING-DAY         PIC 99.
00064              10  TIME-CARD-CLOSING-YR          PIC 99.
00065          05  TIME-CARD-DEPARTMENT          PIC X(4).
00066          05  FILLER                        PIC X(62).
00067
00068      * -----
00069      *
00070      * THE "SORT FILE" MUST BE DEFINED WITH AN "SD" RATHER
00071      * THAN AN "FD".  THE "SD" DEFINES THE RECORD LAYOUT FOR
00072      * THE RECORDS TO BE SORTED.  IT REPRESENTS THE MEMORY
00073      * AREA FROM WHICH THE SORT/MERGE UTILITY PICKS UP ANY
00074      * RECORDS TO BE SORTED, AND TO WHICH THE UTILITY RETURNS
00075      * RECORDS (ONE AT A TIME) AFTER THEY HAVE BEEN REARRANGED.
00076      *
00077      * -----
00078      *
00079      SD  SORTING-TIME-CARD-FILE
00080          RECORD CONTAINS 23 CHARACTERS
00081
00082
00083      01  VALID-TIME-RECORD.
00084          05  VALID-TIME-ID                PIC X(5).
00085          05  VALID-TIME-HOURS             PIC S9(2)V9    COMP-3.
00086          05  VALID-TIME-CLOSING-DATE      PIC X(6).
00087          05  VALID-TIME-DEPARTMENT        PIC X(4).
00088          05  VALID-TIME-EDIT-DATE         PIC X(6).
00089
00090      FD  VALID-TIME-FILE
00091          BLOCK CONTAINS 0 RECORDS
00092          RECORD CONTAINS 23 CHARACTERS
00093          LABEL RECORDS ARE STANDARD

```

شكل (١٢ - ٢) تكملة

```

00094
00095
00096          01  SORTED-VALID-RECORD          PIC X(23).
00097
00098
00099          FD  ERROR-LISTING-FILE
00100              RECORD CONTAINS 132 CHARACTERS
00101              LABEL RECORDS ARE OMITTED
00102
00103
00104          01  ERROR-REPORT-LINE          PIC X(132).
00105
00106  WORKING-STORAGE SECTION.

00215  PROCEDURE DIVISION.
00216
00217  *  -----
00218  *
00219  *  THE SORT VERB MUST NAME THE "SORT FILE" WHICH HAS BEEN
00220  *  PREVIOUSLY DEFINED. IT ALSO INDICATES THE KEY(S) ON
00221  *  WHICH SORTING IS TO BE DONE (THE SORT KEY(S) MUST BE
00222  *  PART OF THE SORT FILE RECORD DESCRIPTION), AND WHETHER
00223  *  THE SORT IS TO BE ASCENDING/DESCENDING ON EACH FIELD.
00224  *  IN THIS CASE THE ENTIRE EDIT PROGRAM IS MADE AN
00225  *  INPUT PROCEDURE TO THE SORT. THE SORTED RECORDS
00226  *  WILL AUTOMATICALLY BE PLACED ON THE "GIVING" FILE.
00227  *
00228  *  -----
00229  *
00230          SORT SORTING-TIME-CARD-FILE
00231              ON ASCENDING KEY VALID-TIME-ID
00232              INPUT PROCEDURE IS 000-EDIT-TIME-CARDS
00233              GIVING VALID-TIME-FILE
00234          STOP RUN
00235
00236
00237  000-EDIT-TIME-CARDS SECTION.
00238
00239          PERFORM 100-INITIALIZE
00240          PERFORM 200-PRODUCE-EDIT-LISTING
00241              UNTIL NO-MORE-RECORDS
00242          PERFORM 300-TERMINATE
00243
00244
00245  PERFORMED-PARAGRAPHS SECTION.
00246
00247  100-INITIALIZE.
00248
00249          OPEN      INPUT  TIME-CARD-FILE
00250                  OUTPUT  ERROR-LISTING-FILE
00251          MOVE "NO" TO          WS-END-TIME-FILE-SW
00252          MOVE ZERO TO          WS-PAGE-NUMBER
00253                              WS-NUMBER-EMPLOYEES
00254                              WS-TOTAL-HOURS
00255          MOVE WS-PAGE-SIZE TO  WS-LINES-ON-PAGE
00256          ACCEPT WS-SYSTEM-DATE FROM DATE
00257          MOVE SYSTEM-YY TO     WS-HEADING-YY
00258          MOVE SYSTEM-MM TO     WS-HEADING-MM
00259          MOVE SYSTEM-DD TO     WS-HEADING-DD

```

```

00260
00261
00262      200-PRODUCE-EDIT-LISTING.
00263
00264      PERFORM 250-GET-TIME-CARD
00265      IF RECORD-AVAILABLE
00266          PERFORM 240-VALIDATE-FIELDS
00267          IF VALID-RECORD
00268              ADD 1 TO WS-NUMBER-EMPLOYEES
00269              ADD TIME-CARD-HOURS TO WS-TOTAL-HOURS
00270              PERFORM 210-RELEASE-TO-SORT
00271          ELSE
00272              PERFORM 220-PRODUCE-ERROR-LISTING
00273
00274
00275      210-RELEASE-TO-SORT.
00276
00277      * -----
00278      *
00279      * RELEASE A RECORD TO THE SORT/MERGE UTILITY FOR SORTING.
00280      * THIS IS DONE BY MOVING THE RECORD INTO THE SORT FILE
00281      * BUFFER AREA, THEN "RELEASING" THE RECORD.
00282      *
00283      * -----
00284
00285      MOVE TIME-CARD-ID TO VALID-TIME-ID
00286      MOVE TIME-CARD-HOURS TO VALID-TIME-HOURS
00287      MOVE TIME-CARD-CLOSING-DATE TO VALID-TIME-CLOSING-DATE
00288      MOVE TIME-CARD-DEPARTMENT TO VALID-TIME-DEPARTMENT
00289      MOVE WS-SYSTEM-DATE TO VALID-TIME-EDIT-DATE
00290      RELEASE VALID-TIME-RECORD
00291
00292
00293      220-PRODUCE-ERROR-LISTING.
00294
00295      MOVE TIME-CARD-ID TO WS-DETAIL-ID
00296      MOVE TIME-CARD-X-HOURS TO WS-DETAIL-HOURS
00297      MOVE TIME-CARD-CLOSING-DATE TO WS-DETAIL-DATE
00298      MOVE TIME-CARD-DEPARTMENT TO WS-DETAIL-DEPARTMENT
00299
00300      MOVE WS-DETAIL-LINE TO WS-OUTPUT-AREA
00301      MOVE WS-SKIP-BEFORE-DETAIL TO WS-NUMBER-TO-SKIP
00302      PERFORM 230-PRINT-OUTPUT-AREA
00303
00304      MOVE WS-UNDER-LINE TO WS-OUTPUT-AREA
00305      MOVE WS-SKIP-BEFORE-UNDER TO WS-NUMBER-TO-SKIP
00306      PERFORM 230-PRINT-OUTPUT-AREA
00307
00308
00309      230-PRINT-OUTPUT-AREA.
00310
00311      IF WS-LINES-ON-PAGE + WS-NUMBER-TO-SKIP
00312          GREATER THAN WS-PAGE-SIZE
00313          ADD 1 TO WS-PAGE-NUMBER
00314          MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE
00315          WRITE ERROR-REPORT-LINE
00316              FROM WS-HEADING-LINE-1
00317              AFTER ADVANCING PAGE
00318          WRITE ERROR-REPORT-LINE
00319              FROM WS-HEADING-LINE-2

```

شكل (١٢ - ٢) تكملة

```

00320      AFTER ADVANCING WS-SKIP-BEFORE-HEADING LINES
00321      MOVE WS-SKIP-BEFORE-HEADING TO WS-LINES-ON-PAGE
00322
00323      WRITE ERROR-REPORT-LINE
00324      FROM WS-OUTPUT-AREA
00325      AFTER ADVANCING WS-NUMBER-TO-SKIP LINES
00326      ADD WS-NUMBER-TO-SKIP TO WS-LINES-ON-PAGE
00327
00328
00329      240-VALIDATE-FIELDS.
00330
00331      MOVE "NO" TO WS-ERROR-DETECTED-SW
00332      MOVE SPACES TO WS-UNDER-LINE
00333      IF TIME-CARD-ID NOT NUMERIC
00334      MOVE ALL "-" TO WS-UNDER-ID
00335      MOVE "YES" TO WS-ERROR-DETECTED-SW
00336
00337      IF TIME-CARD-HOURS NOT NUMERIC
00338      MOVE ALL "-" TO WS-UNDER-HOURS
00339      MOVE "YES" TO WS-ERROR-DETECTED-SW
00340
00341      IF TIME-CARD-CLOSING-DATE NOT NUMERIC
00342      MOVE ALL "-" TO WS-UNDER-DATE
00343      MOVE "YES" TO WS-ERROR-DETECTED-SW
00344      ELSE
00345      PERFORM 260-VALIDATE-DATE.
00346
00347      IF TIME-CARD-DEPARTMENT NOT NUMERIC
00348      MOVE ALL "-" TO WS-UNDER-DEPARTMENT
00349      MOVE "YES" TO WS-ERROR-DETECTED-SW
00350
00351
00352      250-GET-TIME-CARD.
00353
00354      READ TIME-CARD-FILE
00355      AT END
00356      MOVE "YES" TO WS-END-TIME-FILE-SW
00357
00358
00359      260-VALIDATE-DATE.
00360
00361      IF TIME-CARD-CLOSING-YR GREATER THAN SYSTEM-YY
00362      MOVE ALL "-" TO WS-UNDER-YR
00363      MOVE "YES" TO WS-ERROR-DETECTED-SW
00364
00365      IF TIME-CARD-CLOSING-MO LESS THAN 1 OR GREATER THAN 12
00366      MOVE ALL "-" TO WS-UNDER-MO
00367      MOVE "YES" TO WS-ERROR-DETECTED-SW
00368      ELSE IF TIME-CARD-CLOSING-DAY LESS THAN 1 OR
00369      GREATER THAN DAYS-PER-MONTH (TIME-CARD-CLOSING-MO)
00370      MOVE ALL "-" TO WS-UNDER-DAY
00371      MOVE "YES" TO WS-ERROR-DETECTED-SW
00372
00373
00374      300-TERMINATE.
00375
00376      MOVE WS-NUMBER-EMPLOYEES TO WS-FOOTING-COUNT
00377      MOVE WS-TOTAL-HOURS TO WS-FOOTING-TOTAL
00378      MOVE WS-FOOTING-LINE TO WS-OUTPUT-AREA
00379      MOVE WS-SKIP-BEFORE-FOOTING TO WS-NUMBER-TO-SKIP
00380      MOVE WS-FOOTING-LINE TO WS-OUTPUT-AREA
00381      PERFORM 230-PRINT-OUTPUT-AREA
00382
00383      CLOSE TIME-CARD-FILE
00384      ERROR-LISTING-FILE
00385

```

شكل (١٢ - ٢) تكملة

رتب ... مستخدما ... إجراء مخرجات ...

يمكن استخدام رتب ... مستخدما ... إجراء مخرجات SORT... USING... OUTPUT PROCEDURE في الترتيب، وإجراء تشغيل لمحتويات الملف بعد الترتيب، عندما لا تكون هناك حاجة إلى غريفة ، أو تشغيل سابق للسجلات المنطقية (مثل طباعة العناوين البريدية) .

مثال ١٢ - ١٠ :

المسألة ٨ - ٢٢ الخاصة ببرنامج طباعة شيكات لنظام الرواتب ، والتي سبقت معالجة برنامج التنقيح الخاص بها في المسألة ١١ - ٢٢ ، ومثال (١٢ - ٩) ، عدلها لطباعة الشيكات ، مرتبة ترتيبا أبجديا وطبقا لاسم العامل (بدلا من ترتيبها طبقا لرقم العامل) .. ترقب السجلات في PAYROLL - DISK - FILE .
انظر شكل (١٢ - ٣) ؛ حيث تشير سطور التعليقات من ١٥٠ إلى ١٦٠ إلى البرنامج الأصلي، الموجود في خريطة الهيكل في شكل (٨ - ١٧) ، لكنه غير معروض في المسألة ٨ - ٢٢ .

```

00022      SELECT PAYROLL-DISK-FILE
00023          ASSIGN TO PAYDISK
00024          ORGANIZATION IS SEQUENTIAL
00025          ACCESS IS SEQUENTIAL
00026      .
00027      SELECT SORT-CHECK-FILE
00028          ASSIGN TO SORTUTIL
00029      .
00030      SELECT PAYCHECK-FILE
00031          ASSIGN TO CHECKS
00032          ORGANIZATION IS SEQUENTIAL
00033          ACCESS IS SEQUENTIAL
00034      .
00035
00036      DATA DIVISION.
00037
00038      FILE SECTION.
00039
00040      FD  PAYROLL-DISK-FILE
00041          BLOCK CONTAINS 0 RECORDS
00042          RECORD CONTAINS 80 CHARACTERS
00043          LABEL RECORDS ARE STANDARD
00044      .
00045
00046      * -----
00047      * NOTE THAT PAYROLL-CHECK-RECORD NEED NOT BE FULLY
00048      * DESCRIBED WITH SORT...USING...
00049      * -----
00050
00051      01  PAYROLL-CHECK-RECORD          PIC X(80).
00052
00053      SD  SORT-CHECK-FILE
00054          RECORD CONTAINS 80 CHARACTERS
00055
00056
00057      01  SORT-CHECK-RECORD.
00058          05  PAYROLL-ID                PIC X(5).
00059          05  PAYROLL-HOURS             PIC S9(2)V9.
00060          05  PAYROLL-DATE              PIC X(6).

```

شكل (١٢ - ٣)

```

00061      05  PAYROLL-DEPARTMENT      PIC X(4).
00062      05  PAYROLL-RATE             PIC S9(3)V99.
00063      05  PAYROLL-NAME.
00064          10  PAYROLL-INITIALS      PIC X(4).
00065          10  PAYROLL-LAST-NAME      PIC X(16).
00066      05  PAYROLL-GROSS              PIC S9(6)V99.
00067      05  PAYROLL-TAX               PIC S9(5)V99.
00068      05  PAYROLL-NET               PIC S9(6)V99.
00069      05  FILLER                   PIC X(14).
00070
00071      FD  PAYCHECK-FILE
00072          RECORD CONTAINS 132 CHARACTERS
00073          LABEL RECORDS ARE OMITTED
00074
00075
00076      01  PAYCHECK-LINE                PIC X(132).
00077
00078      WORKING-STORAGE SECTION.
.
.
.
00152      PROCEDURE DIVISION.
00153
00154      *  -----
00155      *  THE SORT STATEMENT IS PLACED IN THE HIGHEST LEVEL
00156      *  MODULE.  THE REST OF THE ORIGINAL PROGRAM
00157      *  BECOMES THE OUTPUT PROCEDURE, WITH ONLY THE FOLLOWING
00158      *  CHANGES NECESSARY:  1) PAYROLL-DISK-FILE MUST NOT BE
00159      *  OPENED SINCE IT IS THE USING FILE, AND 2) "READ
00160      *  PAYROLL-DISK-FILE" BECOMES "RETURN SORT-CHECK-FILE"
00161      *  -----
00162
00163          SORT SORT-CHECK-FILE
00164              ON ASCENDING KEY PAYROLL-LAST-NAME
00165              USING PAYROLL-DISK-FILE
00166              OUTPUT PROCEDURE IS PRINT-STUBS-AND-CHECKS
00167          STOP RUN
00168
00169
00170      PRINT-STUBS-AND-CHECKS SECTION.
00171
00172          PERFORM 100-INITIALIZE
00173          PERFORM 200-PRODUCE-PAYCHECKS
00174              UNTIL NO-MORE-RECORDS
00175          PERFORM 300-TERMINATE
00176
00177      *-----
00178      PERFORMED-PARAGRAPHS SECTION.
00179      *-----
00180      100-INITIALIZE.
00181
00182          OPEN      OUTPUT  PAYCHECK-FILE
00183          MOVE "NO" TO      WS-END-DISK-FILE-SW
00184          MOVE ZERO TO      WS-NUMBER-EMPLOYEES
00185                          WS-TOTAL-HOURS
00186          ACCEPT WS-TODAYS-DATE FROM DATE
00187          MOVE SYSTEM-YY      TO WS-YY
00188          MOVE SYSTEM-MM      TO WS-MM
00189          MOVE SYSTEM-DD      TO WS-DD
00190          ACCEPT WS-CHECK-NUMBER
00191          FROM STARTING-CHECK-NUMBER-DEVICE

```

```

00192
00193     PERFORM 110-PRODUCE-TEMPLATE
00194     PERFORM 210-GET-NEXT-EMPLOYEE
00195
00196
00197     110-PRODUCE-TEMPLATE.
00198
00199         MOVE ALL "*" TO WS-STUB-ID
00200                     WS-STUB-NAME
00201                     WS-STUB-DATE
00202                     WS-CHECK-NAME
00203     PERFORM 230-PRINT-CHECK
00204         WS-NUMBER-OF-TEMPLATES TIMES
00205
00206
00207     200-PRODUCE-PAYCHECKS.
00208
00209         PERFORM 220-FORMAT-CHECKS
00210         PERFORM 230-PRINT-CHECK
00211         ADD PAYROLL-HOURS             TO WS-TOTAL-HOURS
00212         ADD 1                         TO WS-NUMBER-EMPLOYEES
00213                                     WS-CHECK-NUMBER
00214     • PERFORM 210-GET-NEXT-EMPLOYEE
00215
00216
00217     210-GET-NEXT-EMPLOYEE.
00218
00219         RETURN SORT-CHECK-FILE
00220         AT END
00221         MOVE "YES" TO WS-END-DISK-FILE-SW
00222
00223
00224     220-FORMAT-CHECKS.
00225
00226         MOVE PAYROLL-ID             TO WS-STUB-ID
00227         MOVE PAYROLL-NAME           TO WS-STUB-NAME
00228                                     WS-CHECK-NAME
00229         MOVE PAYROLL-DEPARTMENT     TO WS-STUB-DEPARTMENT
00230         MOVE PAYROLL-DATE           TO WS-STUB-DATE
00231         MOVE PAYROLL-HOURS          TO WS-STUB-HOURS
00232         MOVE PAYROLL-RATE           TO WS-STUB-RATE
00233         MOVE PAYROLL-GROSS          TO WS-STUB-GROSS
00234         MOVE PAYROLL-TAX            TO WS-STUB-TAX
00235         MOVE PAYROLL-NET            TO WS-STUB-NET
00236                                     WS-CHECK-AMOUNT
00237
00238
00239     230-PRINT-CHECK.
00240
00241         WRITE PAYCHECK-LINE
00242             FROM WS-STUB-LINE-1
00243             AFTER ADVANCING WS-SKIP-BEFORE-STUB-1 LINES
00244         WRITE PAYCHECK-LINE
00245             FROM WS-STUB-LINE-2
00246             AFTER ADVANCING WS-SKIP-BEFORE-STUB-2 LINES
00247         WRITE PAYCHECK-LINE
00248             FROM WS-CHECK-LINE-1
00249             AFTER ADVANCING WS-SKIP-BEFORE-CHECK-1 LINES
00250         WRITE PAYCHECK-LINE
00251             FROM WS-CHECK-LINE-2
00252             AFTER ADVANCING WS-SKIP-BEFORE-CHECK-2 LINES
00253
00254
00255     300-TERMINATE.
00256
00257         CLOSE    PAYCHECK-FILE
00258

```

شكل (١٢ - ٣) تكملة

رتب ... إجراء مدخلات ... إجراء مخرجات

مثال ١٢ - ١١ :

نوضح في شكل (١٢ - ٤) كلاً من إجراء المدخلات وإجراء المخرجات مع برنامج إنتاج ملف مفهرس indexed file ، وهو ملف شائع الاستخدام جداً في التطبيقات التي تحتاج إلى تشغيل ملفات عشوائياً . الطريقة الأكثر كفاءة لإنتاج ملفات مفهرسة هي بنائها تتابعياً ، ووضع السجلات في الملف : مرتبة طبقاً لأحد المفاتيح . ويستخدم إجراء المدخلات في اكتشاف سجلات العمليات الجارية غير الصحيحة ، ومنعها من الترتيب ، ثم ينتج إجراء المخرجات بعد ذلك الملف المفهرس فعلياً . لاحظ أن إجراء المدخلات وإجراء المخرجات يشبهان برنامجين منفصلين ؛ كل منهما يفتح ملفات ويضع قيماً ابتدائية لمفاتيح ، ثم ينفذ مقطع تشغيل رئيسي ، حتى تنتهي البيانات ، فيغلق الملفات وينهي العمل . ليس هذا مدهشاً ؛ لأن الجزئين ينفذان منفصلان عن بعضهما البعض تحت تحكم فعل SORT .

```

00015      INPUT-OUTPUT SECTION.
00016
00017      FILE-CONTROL.
00018
00019          SELECT CREATION-INFO-FILE
00020              ASSIGN TO CREATE
00021              ORGANIZATION IS SEQUENTIAL
00022              ACCESS IS SEQUENTIAL
00023
00024          SELECT SORTED-TRANS-FILE
00025              ASSIGN TO SORTWK
00026
00027          SELECT PRICING-MASTER-FILE
00028              ASSIGN TO PRICES
00029              ORGANIZATION IS INDEXED
00030              ACCESS IS SEQUENTIAL
00031              RECORD KEY IS PRICING-ITEM-ID
00032              FILE STATUS IS MASTER-STATUS-CODE
00033
00034          SELECT ERROR-LOG
00035              ASSIGN TO ERRORS
00036              ORGANIZATION IS SEQUENTIAL
00037              ACCESS IS SEQUENTIAL
00038
00039
00040      DATA DIVISION.
00041
00042      FILE SECTION.
00043
00044      FD CREATION-INFO-FILE
00045          RECORD CONTAINS 80 CHARACTERS
00046          LABEL RECORDS ARE OMITTED
00047
00048      01 CREATION-RECORD.
00049          05 CREATION-ITEM-ID          PIC X(5).
00050          05 CREATION-ITEM-DESCRIPTION
00051              PIC X(25).
00052          05 CREATION-PRICE            PIC 9(5)V99.
00053          05 CREATION-PRICE-X          REDEFINES CREATION-PRICE
00054              PIC X(7).
00055          05 CREATION-QUANTITY         PIC 9(5).
00056          05 CREATION-QUANTITY-X      REDEFINES CREATION-QUANTITY

```

شكل (١٢ - ٤)

```

00057                                PIC X(5).
00058      05 FILLER                      PIC X(38).
00059
00060      SD SORTED-TRANS-FILE
00061         RECORD CONTAINS 80 CHARACTERS
00062
00063      01 TRANS-RECORD.
00064         05 TRANSACT-ITEM-ID          PIC X(5).
00065         05 TRANSACT-ITEM-DESCRIPTION PIC X(25).
00066         05 TRANSACT-PRICE            PIC S9(5)V99.
00067         05 TRANSACT-PRICE-X          REDEFINES TRANSACT-PRICE
00068                                         PIC X(7).
00069         05 TRANSACT-QUANTITY         PIC S9(5).
00070         05 TRANSACT-QUANTITY-X       REDEFINES TRANSACT-QUANTITY
00071                                         PIC X(5).
00072         05 FILLER                     PIC X(38).
00073
00074
00075      FD PRICING-MASTER-FILE
00076         RECORD CONTAINS 37 CHARACTERS
00077         LABEL RECORDS ARE STANDARD
00078
00079      01 PRICING-RECORD.
00080         05 PRICING-ITEM-ID          PIC X(5).
00081         05 PRICING-ITEM-DESCRIPTION PIC X(25).
00082         05 PRICING-QUANTITY-ONHAND PIC S9(5)      COMP-3.
00083         05 PRICING-PRICE            PIC S9(5)V99  COMP-3.
00084
00085
00086      FD ERROR-LOG
00087         RECORD CONTAINS 132 CHARACTERS
00088         LABEL RECORDS ARE OMITTED
00089         LINAGE IS 60 LINES
00090             WITH FOOTING AT 59
00091             LINES AT TOP 3
00092             LINES AT BOTTOM 3
00093
00094
00095      01 ERROR-LINE                      PIC X(132).
00096
00097      WORKING-STORAGE SECTION.
00098
00099      01 PROGRAM-SWITCHES.
00100         05 WS-END-CREATION-SW        PIC X(3).
00101             88 NO-MORE-RECORDS      VALUE "YES".
00102             88 MORE-RECORDS         VALUE "NO".
00103         05 WS-VALID-TRANS-SW        PIC X(3).
00104             88 VALID-TRANS          VALUE "YES".
00105             88 INVALID-TRANS        VALUE "NO".
00106         05 MASTER-STATUS-CODE       PIC XX.
00107
00108      01 PROGRAM-COUNTERS.
00109         05 WS-PAGE-NUMBER            PIC S9(3)      COMP-3.
00110
00111      01 SYSTEM-DATE-AREA.
00112         05 SYSTEM-YY                PIC 99.
00113         05 SYSTEM-MM                PIC 99.
00114         05 SYSTEM-DD                PIC 99.
00115

```

```

00116      01 PROGRAM-CONSTANTS.
00117      05 WS-MESSAGES.
00118          10 WS-INVALID-FIELDS-MSG          PIC X(30)
00119              VALUE "INVALID FIELDS".
00120          10 WS-FILE-FULL-MSG          PIC X(30)
00121              VALUE "FILE FULL--NO ADD".
00122          10 WS-DUPLICATE-MESSAGE      PIC X(30)
00123              VALUE "DUPLICATE KEY--NO ADD".
00124
00125
00126      01 WS-HEADING-LINE-1.
00127          05 FILLER          PIC X(3)          VALUE SPACES.
00128          05 FILLER          PIC X(32)
00129              VALUE "PRICING FILE CREATION ERROR LOG".
00130          05 WS-MM          PIC Z9.
00131          05 FILLER          PIC X          VALUE "/".
00132          05 WS-DD          PIC 99.
00133          05 FILLER          PIC X          VALUE "/".
00134          05 WS-YY          PIC 99.
00135          05 FILLER          PIC X(2)          VALUE SPACES.
00136          05 FILLER          PIC X(5)          VALUE "PAGE ".
00137          05 WS-HEADING-PAGE      PIC ZZ9.
00138          05 FILLER          PIC X(79)          VALUE SPACES.
00139
00140      01 WS-HEADING-LINE-2.
00141          05 FILLER          PIC X(10)          VALUE "ITEM-ID".
00142          05 FILLER          PIC X(27)
00143              VALUE "DESCRIPTION".
00144          05 FILLER          PIC X(9)          VALUE "PRICE".
00145          05 FILLER          PIC X(86)          VALUE "QUANTITY".
00146
00147      01 WS-DETAIL-LINE.
00148          05 WS-DETAIL-ITEM-ID      PIC X(5).
00149          05 FILLER          PIC X(5)          VALUE SPACES.
00150          05 WS-DETAIL-DESCRIPTION  PIC X(25).
00151          05 FILLER          PIC X(2)          VALUE SPACES.
00152          05 WS-DETAIL-PRICE        PIC X(7).
00153          05 FILLER          PIC X(2)          VALUE SPACES.
00154          05 WS-DETAIL-QUANTITY     PIC X(5).
00155          05 FILLER          PIC X(2)          VALUE SPACES.
00156          05 WS-DETAIL-MESSAGE     PIC X(79).
00157
00158      PROCEDURE DIVISION.
00159
00160          SORT SORTED-TRANS-FILE
00161              ON ASCENDING KEY TRANSACT-ITEM-ID
00162              INPUT PROCEDURE IS 000-SCREEN-TRANSACTIONS
00163              OUTPUT PROCEDURE IS 001-CREATE-PRICING-MASTER
00164          STOP RUN
00165
00166
00167      000-SCREEN-TRANSACTIONS SECTION.
00168
00169          OPEN OUTPUT ERROR-LOG
00170          OPEN INPUT CREATION-INFO-FILE
00171          MOVE "NO" TO WS-END-CREATION-SW
00172

```

```

00173      ACCEPT SYSTEM-DATE-AREA FROM DATE
00174      MOVE SYSTEM-MM TO WS-MM
00175      MOVE SYSTEM-DD TO WS-DD
00176      MOVE SYSTEM-YY TO WS-YY
00177
00178      MOVE ZERO TO WS-PAGE-NUMBER
00179
00180      PERFORM 170-PRINT-HEADINGS
00181      PERFORM 180-GET-NEXT-TRANS
00182
00183      PERFORM 100-GENERATE-PRICING-RECORD
00184      UNTIL NO-MORE-RECORDS
00185
00186      CLOSE CREATION-INFO-FILE
00187
00188
00189      001-CREATE-PRICING-MASTER SECTION.
00190
00191      OPEN OUTPUT PRICING-MASTER-FILE
00192      MOVE "NO" TO WS-END-CREATION-SW
00193      PERFORM 140-RETURN-NEXT-TRANS
00194      PERFORM 120-PRODUCE-MASTER-REC
00195      UNTIL NO-MORE-RECORDS
00196      CLOSE PRICING-MASTER-FILE
00197      ERROR-LOG
00198
00199
00200      PERFORMED-PARAGRAPHS SECTION.
00201
00202      100-GENERATE-PRICING-RECORD.
00203
00204      PERFORM 110-VALIDATE-TRANS
00205      IF VALID-TRANS
00206      PERFORM 125-RELEASE-TO-SORT
00207      ELSE
00208      MOVE WS-INVALID-FIELDS-MSG TO WS-DETAIL-MESSAGE
00209      PERFORM 130-PRODUCE-ERROR-LOG
00210
00211      PERFORM 180-GET-NEXT-TRANS
00212
00213
00214      110-VALIDATE-TRANS.
00215
00216      IF CREATION-PRICE NOT NUMERIC
00217      OR CREATION-QUANTITY NOT NUMERIC
00218      MOVE "NO" TO WS-VALID-TRANS-SW
00219      ELSE
00220      MOVE "YES" TO WS-VALID-TRANS-SW
00221
00222
00223      120-PRODUCE-MASTER-REC.
00224
00225      MOVE TRANSACT-ITEM-ID TO PRICING-ITEM-ID
00226      MOVE TRANSACT-ITEM-DESCRIPTION
00227      TO PRICING-ITEM-DESCRIPTION
00228      MOVE TRANSACT-PRICE TO PRICING-PRICE
00229      MOVE TRANSACT-QUANTITY TO PRICING-QUANTITY-ONHAND
00230      PERFORM 150-WRITE-MASTER-RECORD
00231      IF MASTER-STATUS-CODE NOT EQUAL "00"

```

```

00232             MOVE WS-DUPLICATE-MESSAGE    TO WS-DETAIL-MESSAGE
00233             PERFORM 130-PRODUCE-ERROR-LOG
00234
00235             PERFORM 140-RETURN-NEXT-TRANS
00236
00237
00238             125-RELEASE-TO-SORT.
00239
00240             RELEASE TRANS-RECORD
00241             FROM CREATION-RECORD
00242
00243
00244             130-PRODUCE-ERROR-LOG.
00245
00246             MOVE TRANSACT-ITEM-ID            TO WS-DETAIL-ITEM-ID
00247             MOVE TRANSACT-ITEM-DESCRIPTION  TO WS-DETAIL-DESCRIPTION
00248
00249             MOVE TRANSACT-PRICE-X           TO WS-DETAIL-PRICE
00250             MOVE TRANSACT-QUANTITY-X        TO WS-DETAIL-QUANTITY
00251             PERFORM 160-PRINT-DETAIL-LINE
00252
00253
00254             140-RETURN-NEXT-TRANS.
00255
00256             RETURN SORTED-TRANS-FILE
00257             AT END
00258             MOVE "YES" TO WS-END-CREATION-SW
00259
00260
00261
00262             150-WRITE-MASTER-RECORD.
00263
00264             WRITE PRICING-RECORD
00265
00266
00267             160-PRINT-DETAIL-LINE.
00268
00269             WRITE ERROR-LINE
00270             FROM WS-DETAIL-LINE
00271             AFTER ADVANCING 2 LINES
00272             AT END-OF-PAGE
00273             PERFORM 170-PRINT-HEADINGS
00274
00275
00276             170-PRINT-HEADINGS.
00277
00278             ADD 1 TO WS-PAGE-NUMBER
00279             MOVE WS-PAGE-NUMBER              TO WS-HEADING-PAGE
00280             WRITE ERROR-LINE
00281             FROM WS-HEADING-LINE-1
00282             AFTER ADVANCING PAGE
00283             WRITE ERROR-LINE
00284             FROM WS-HEADING-LINE-2
00285             AFTER ADVANCING 2 LINES
00286
00287
00288             180-GET-NEXT-TRANS.
00289
00290             READ CREATION-INFO-FILE
00291             AT END
00292             MOVE "YES" TO WS-END-CREATION-SW
00293
00294

```

شكل (١٢ - ٤) تكملة

ترتيب مجموعة عناصر

يمكن أن تتحسن كفاءة الترتيب طبقاً لعدة مفاتيح بتجميع حقول المفاتيح مع بعضها البعض، وإجراء الترتيب طبقاً لمجموعة العناصر. ويمكن تطبيق هذه الطريقة عندما تكون المفاتيح المطلوب تجميعها:

(١) متجاوزة داخل السجل.

(٢) حرفية عددية أو عددية بدون إشارة مع استخدام DISPLAY.

(٣) كلها تصاعدياً أو كلها تنازلياً.

(٤) مرتبة من الحقل الرئيسى الى الحقل الأدنى ؛ حيث يكون الحقل الرئيسى أول عنصر فى المجموعة ، والحقل الأدنى

آخر عنصر فى المجموعة.

مثال ١٢ - ١٢ :

```
SD SORT-FILE ...
01 SORT-RECORD.
   05 SORT-KEYS.
      10 REGION-ID          PIC X(4).
      10 DEPARTMENT-ID      PIC X(7).
      10 SALESPERSON-ID     PIC X(3).
   05 REGION-NAME          PIC X(10).
   05 DEPARTMENT-NAME      PIC X(20).
   05 SALESPERSON-NAME     PIC X(15).
.....
SORT SORT-FILE
ON ASCENDING KEY SORT-KEYS
.....
```

١٢ - ٣ دمج الملفات بفعل الدمج

الدمج merging هو خلط ملفين مرتبين فعلاً (أو أكثر) مع بعضهما فى ملف واحد ، ويمكن الكلام عن دمج n طريقة (n -way merge) للإشارة الى دمج n ملف فى ملف واحد . وفى كويل IBM OS/VS .. يجب أن تكون $n \leq 8$. وتعالج نفس منفعة النظام التى تجرى ترتيب الملفات ، فى العادة - دمج الملفات.

وحيث يشمل كل من الترتيب والدمج ترتيباً ordering للسجلات فى ملفات .. فإن مفهوم الحقول الرئيسىة ، أو حقول المفاتيح والتتابع التصاعدي أو التنازلي، والمفاتيح : الرئيسى والمتوسط والأدنى هى نفسها فى كلتا الحالتين.

يشبه تكوين عبارة الدمج MERGE شكل (١٢ - ٥) تكوين عبارة الترتيب . ولعمل أى منها .. يجب أن يعرف المبرمج ملف ترتيب ، أو دمجاً خاصاً بالصورة المقيدة لعبارة SELECT فى جزء الأوساط ، وباستخدام SD فى جزء البيانات. ويعيداً عن شكل عبارات SORT, MERGE نفسها .. فإن متطلبات الكويل لعمل الترتيب والدمج لا تتغير .

يجب أن تعرف عبارة الدمج MERGE حقول المفاتيح التى رتب على أساسها الملفات ، المراد دمجها ، والتى على أساسها يرتب الملف المدمج. وتكتب المفاتيح فى تسلسل من المفتاح الرئيسى (أولاً) الى المفتاح الأدنى (أخيراً)، كما هو الحال تماماً فى عبارة SORT .

```

MERGE file-name-1
  ON { ASCENDING } KEY data-name-1 [data-name-2] ...
     { DESCENDING }
[ ON { ASCENDING } KEY data-name-3 [data-name-4] ...
   { DESCENDING }
[ COLLATING SEQUENCE IS alphabet-name ]
  USING file-name-2 file-name-3 [file-name-4] ...
  { GIVING file-name-5
    OUTPUT PROCEDURE IS section-name-1 { { THROUGH } section-name-2 |
                                           { THRU }
  }

```

شكل (١٢ - ٥)

لاحظ أنه غير مسموح باستخدام INPUT PROCEDURE في عبارة MERGE .. يجب أن تغلق ملفات USING (المراد دمجها طبقا لمواصفات ASCENDING / DESCENDING KEY) . عندما تنفذ عبارة MERGE ؛ حيث تفتحها عبارة MERGE تلقائيا . وعندما تنتهي عملية الدمج .. تغلق MERGE الملفات تلقائيا .

يحدد جزء GIVING أن السجلات المدمجة تكتب في الملف المحدد مباشرة ؛ ويجب أن يكون ملف GIVING مغلقا عندما تنفذ MERGE ؛ حيث تفتح منفعة الترتيب أو الدمج الملفات تلقائيا ، وتكتب السجلات المدمجة في الملف ، ثم تغلق الملف مرة أخرى .

يعمل إجراء المخرجات OUTPUT PROCEDURE مثل نظيره في ملف عبارة SORT تماما .

File A	File B	File C	Merged File
20	10	5	5
40	30	25	10
60	50	35	20
80	70		25
	90		30
			35
			40
			50
			60
			70
			80
			90

شكل (١٢ - ٦)

مثال ١٢ - ١٢ :

لدمج الملفات المبينة في شكل (٢ - ٦) .. فإننا نكتب الشفرة التالية :

ENVIRONMENT DIVISION.

```
.....
SELECT FILE-A ASSIGN TO...
SELECT FILE-B ASSIGN TO...
SELECT FILE-C ASSIGN TO...
SELECT MERGE-FILE
ASSIGN TO SORTWK.
.....
```

DATA DIVISION.

FILE SECTION.

```
FD FILE-A...
01 RECORD-A...
```

```
FD FILE-B...
01 RECORD-B...
```

```
FD FILE-C
01 RECORD-C...
```

```
SD MERGE-FILE
RECORD CONTAINS...
01 MERGE-RECORD.
05 MERGE-KEY-1...
05 DATA-FIELD-1...
05 MERGE-KEY-2...
05 DATA-FIELD-2...
.....
```

PROCEDURE DIVISION.

```
MERGE MERGE-FILE
ON ASCENDING KEY MERGE-KEY-1
ON DESCENDING KEY MERGE-KEY-2
USING FILE-A FILE-B FILE-C
OUTPUT PROCEDURE IS PROCESS-MERGED-RECORDS
STOP RUN
```

PROCESS-MERGED-RECORDS SECTION.

PERFORMED-PARAGRAPHS SECTION.

```
GET-MERGED-RECORD.
RETURN MERGE-FILE
AT END...
```


يظل هذا البرنامج صحيحا مع استبدال عبارة MERGE بعبارة SORT ، يمكن أن يكون هذا الاستبدال ضروريا إذا لم تكن الملفات المراد ترتيبها مرتبة فعلا ، وحيث إنها مرتبة .. فيصبح فعل MERGE أكثر كفاءة بالطبع عن فعل SORT .

لاحظ أنه إذا تحدد أكثر من ملف USING واحد في عبارة SORT .. تخطط كل السجلات من كل الملفات وترتب ، وعلى هذا .. يكون لدينا أسلوب لدمج ملفات غير مرتبة.

أسئلة مراجعة

- ١٢ - ١ ميز بين ترتيب ملف وترتيب جدول.
- ١٢ - ٢ وضع المقصود : حقل مفتاح ، ترتيب تصاعدي ، ترتيب تنازلي ، مفتاح رئيسي ، مفتاح متوسط ، مفتاح أدنى.
- ١٢ - ٣ وضع العلاقة بين منفعة الترتيب ومنفعة الدمج للنظام من ناحية ، وعبارة SORT وعبارة MERGE من ناحية أخرى .
- ١٢ - ٤ ما المقصود بملف الترتيب ؟
- ١٢ - ٥ اذكر قيود عبارة SELECT وأجزاء FD عند تعريف ملف ترتيب ، أو ملف دمج ..
- ١٢ - ٦ وضع الغرض من عبارة RELEASE وعبارة RETURN .
- ١٢ - ٧ اذكر عدة أمثلة يمكن أن يكون فيها إجراء المدخلات مفيدا .
- ١٢ - ٨ اذكر عدة أمثلة يمكن أن يكون فيها إجراء المخرجات مفيدا .
- ١٢ - ٩ اذكر مثالا يمكن أن يبنى إجراء المدخلات فيه السجلات التي يزيحها إلى منفعة الترتيب أو الدمج .
- ١٢ - ١٠ اذكر قيود مفاتيح الترتيب عندما ترتب سجلات متغيرة الطول .
- ١٢ - ١١ وضع الغرض من جزء تسلسل التابع في عبارات SORT, MERGE .
- ١٢ - ١٢ لماذا لا يشيع ترتيب SORT... GIVING في برامج الكويل ؟
- ١٢ - ١٣ ناقش هيكل جزء الإجراءات عند استخدام عبارة SORT ، أو عبارة MERGE .
- ١٢ - ١٤ ناقش سلسلة الأحداث التي تحدث عند تنفيذ عبارة SORT .
- ١٢ - ١٥ ماذا يحدث إذا نفذت العبارة STOP RUN داخل إجراء مدخلات أو داخل إجراء مخرجات ؟
- ١٢ - ١٦ ناقش العلاقة بين إجراء المدخلات وإجراء المخرجات تحت تحكم أمر SORT .
- ١٢ - ١٧ وضع كيف يحسن تجميع مفاتيح الترتيب من كفاءة الترتيب . ما القيود التي تقع على هذا الأسلوب ؟
- ١٢ - ١٨ ميز بين ترتيب الملف ودمج الملفات .
- ١٢ - ١٩ اذكر بعض الأمثلة للمواقف التي يمكن فيها استخدام دمج الملفات ..
- ١٢ - ٢٠ ماذا يعني دمج ثلاث طرق three - way merge ؟
- ١٢ - ٢١ لماذا يجب إغلاق ملفات GIVING, USING ، قبل تنفيذ SORT ، أو MERGE ؟
- ١٢ - ٢٢ كيف تترجم عملية الدمج إذا لم تكن الملفات المطلوب دمجها مرتبة ؟

مسائل محلولة

- ١٢ - ٢٣ (١) عرف المفتاح الرئيسي ، والمتوسط ، والأدنى في حقول السجل التالي .

(ب) أى المفاتيح يكون تصاعدياً ، وأياً يكون تنازلياً ؟

Name	Region	Number-of-Accounts
PERELMAN	10	5
SNYDER	10	5
ABEL	10	3
CONVERSE	10	3
FOLKERS	20	8
GETZ	20	8
BAKER	20	4

(١) الحقل الرئيسى : region

الحقل المتوسط : number - of - accounts

الحقل الأدنى : name

(ب) name, Region فى ترتيب تصاعدي ، بينما number - of - accounts فى ترتيب تنازلى .

١٢ - ٢٤ بالإضافة إلى منفعة الترتيب والدمج.. ما برامج المنفعة الأخرى المتاحة تقليدياً فى نظام الكمبيوتر ؟

برامج لنسخ الملفات، وإطباعة محتويات ملف، وإسرد كل الملفات على مجموعة أقراص معينة أو على شريط ؛ لمحاولة استعادة ملفات من مجموعة أقراص ، أو شريط أصابها التلف ... إلخ .

١٢ - ٢٥ ما الفرق بين تعريفات الكويل ملف ترتيب و ملف دمج ؟

لا يوجد فرق ؛ إذ يحتاج كل منهما إلى صيغة محددة لعبارة SELECT ، ويعرف كل منهما على مستوى SD .

لا توجد طريقة تذكر إذا كان ملف الترتيب مستخدماً مع SORT ، أو مع MERGE ، وذلك باستثناء فحص العبارات الفعلية لجزء الإجراءات .

١٢ - ٢٦ ما الصيغة المناسبة لاسم ASSIGN TO ملف ترتيب أو دمج ؟

أى اسم صحيح ؛ حيث يعامل اسم ASSIGN TO كتعليق .

١٢ - ٢٧ اذكر ما إذا كان ما يلى يشمل : USING ، أو INPUT PROCEDURE ، أو GIVING ، أو OUTPUT PROCEDURE : CEDURE

(أ) برنامج يطبع قائمة بعناوين بريديّة لكل السجلات الموجودة فى شريط عناوين العملاء البريديّة .

(ب) برنامج يحسب ويطبّع متوسط نقاط درجات الطلبة الدارسين (نون الحصول على مؤهل جامعي) بأحد الجامعات .

(ج) برنامج يعد نسخة احتياطية من ملف شيكات حسابات الدائنين ، يجب أن يرتب طبقاً لرقم دفتر الأستاذ العام ؛ بدلاً من ترتيبه طبقاً لرقم الشيك .

(د) برنامج تنقيح ينتج ملف قرص لعمليات جارية مرتبة ، وذلك بعد التأكد من صحتها .

(١) USING مع OUTPUT PROCEDURE (لتشكيل تقرير الطباعة) .

(ب) INPUT PROCEDURE (لاختيار الطلبة غير الدراسين ؛ للحصول على مؤهل جامعي) مع OUTPUT PROCEDURE (لحساب متوسط النقاط) .

(ج) USING مع GIVING (في الحياة الواقعية ... يمكن تشغيل منفعة الترتيب أو الدمج كبرنامج قائم بذاته ؛ بدلا من استدعائه من خلال برنامج الكويل) .

(د) INPUT PROCEDURE (لغزلة العمليات الجارية غير الصحيحة) مع GIVING .

١٢ - ٢٨ حدد الخطأ فيما يلي :

SD ANY-SORT-FILE
BLOCK CONTAINS 10 RECORDS
RECORD CONTAINS 200 CHARACTERS
LABEL RECORDS ARE STANDARD

يسمح بالأجزاء DATA RECORDS ARE , RECORD CONTAINS فقط في تعريف ملف الترتيب أو الدمج (SD).
لا يعد الجزأان BLOCK CONTAINS, LABEL RECORDS صحيحين .

١٢ - ٢٩ اكتب برنامجا يرتب سجلات مبيعات ؛ طبقا لمتوسط المبيعات العميل ، كما يحسب من إجراء المدخلات .

SELECT SALES-FILE ASSIGN TO ...
SELECT SORT-FILE ASSIGN TO ANYTHING.

```
.....
FD SALES-FILE ...
01 SALES-RECORD.
   05 SALESPERSON-ID          PIC X(7).
   05 TOTAL-SALES             PIC S9(5)V99  COMP.
   05 NUMBER-CUSTOMERS        PIC S9(3)     COMP.
SD SORT-FILE
  RECORD-CONTAINS ...
01 SORT-RECORD.
   05 SORT-SALES-ID          PIC X(7).
   05 SORT-TOTAL-SALES       PIC S9(5)V99  COMP.
   05 SORT-NUMBER-CUSTOMERS  PIC S9(3)     COMP.
   05 SORT-AVERAGE-SALES    PIC S9(5)V99  COMP.
.....
```

SORT SORT-FILE
ON DESCENDING KEY SORT-AVERAGE-SALES
INPUT PROCEDURE IS CALCULATE-AVERAGE
OUTPUT PROCEDURE IS PRINT-REPORT
STOP RUN

CALCULATE-AVERAGE SECTION.
OPEN INPUT SALES-FILE
MOVE "NO" TO END-FILE-SW
PERFORM GET-SALES-RECORD
PERFORM RELEASE-A-RECORD
UNTIL END-FILE-SW EQUAL "YES"

CLOSE SALES-FILE

PRINT-REPORT SECTION.

PERFORMED-PARAGRAPHS SECTION.

GET-SALES-RECORD.

READ SALES-FILE

AT END MOVE "YES" TO END-FILE-SW

RELEASE-A-RECORD.

MOVE SALESPERSON-ID TO SORT-SALES-ID

MOVE TOTAL-SALES TO SORT-TOTAL-SALES

MOVE NUMBER-CUSTOMERS TO SORT-NUMBER-CUSTOMERS

DIVIDE TOTAL-SALES BY NUMBER-CUSTOMERS

GIVING SORT-AVERAGE-SALES

RELEASE SORT-RECORD

PERFORM GET-SALES-RECORD

١٢ - ٣٠ يضع تسلسل التتابع المعتاد EBCDIC الأرقام من 0 إلى 9 بعد الحروف من A إلى Z . قم بعمل ترتيب ؛ يضع الأرقام من 0 إلى 9 قبل الحروف من A إلى Z .

SPECIAL-NAMES.

CUSTOMER-SORT-ORDER IS "0" THRU "9"

"A" THRU "Z"

SORT SORT-FILE

ON ASCENDING KEY WHATEVER

COLLATING SEQUENCE IS CUSTOMER-SORT-ORDER

INPUT PROCEDURE IS SCREEN-THEM

OUTPUT PROCEDURE IS PROCESS-THEM

١٢ - ٣١ ناقش كيف يمكن ترتيب حقول تاريخ .

يمكن أن ترتب التواريخ المعرفة بالصورة (6) PIC X أو (6) PIC 9 بطريقة صحيحة إذا كانت الستة رموز في حقل التاريخ مرتبة على النحو التالي yymmdd (yy هي النسبة ، و mm الشهر ، و dd اليوم) بدلا من الشكل المعتاد mmdyy . يحتفظ الشكل yymmdd بالترتيب المناسب لمفاتيح الترتيب من الرئيسي إلى الأدنى داخل حقل التاريخ . والطريقة الأخرى هي

استخدام التاريخ برقم اليوم فى السنة yyddd (القسم السادس - الفصل السادس) . لا تنتج أى من هاتين الطريقتين نتائج صحيحة إذا تغير القرن.

١٢ - ٢٢ لا تسمح بعض مترجمات الكويل بإجراء مداخلات ، أو إجراء مخرجات لتنفيذ مقاطع ، تقع خارج قسم المداخلات أو قسم المخرجات المناسب .

باقتراض استخدام مثل هذا المترجم .. أعد كتابة إجراء مداخلات المسألة ٢٩ .

```
CALCULATE-AVERAGE SECTION.
  OPEN INPUT SALES-FILE
  MOVE "NO" TO END-FILE-SW
  PERFORM GET-SALES-RECORD
  PERFORM RELEASE-A-RECORD
  UNTIL END-FILE-SW EQUAL "YES"
  CLOSE SALES-FILE
  GO TO CALCULATE-AVERAGE-EXIT
```

GET-SALES-RECORD.

RELEASE-A-RECORD.

CALCULATE-AVERAGE-EXIT.

EXIT.

PRINT-REPORT SECTION.

تعد عبارة GO TO عبارة واسعة الاستخدام فى كويل غير المرتب ، وتكون هناك حاجة لهذه العبارة فى إعداد الشفرة السابقة لمنع GET - SALES - RECORD ، و RELEASE - A - RECORD الموجودين الآن فى SECTION من التنفيذ مره اخرى بعد إغلاق SALES - FILE . لا تفعل عبارة EXIT شيئاً ، ولكنها تسمح بأن يعمل CALCULATE - AVERAGE - EXIT كمقطع إنهاء للقسم SECTION .

١٢ - ٢٣ أعد تعريف سجل الترتيب التالى لعمل ترتيب أكثر كفاءة .

```
SD SORT-FILE...
01 SORT-REC.
   05 EMPLOYEE-ID          PIC X(4).
   05 NAME                  PIC X(20).
   05 DEPT                  PIC XX.
   05 NUMBER-DEPENDENTS     PIC S9          COMP-3.
   05 HOURLY-RATE           PIC S9(2)V99    COMP.
```

SORT SORT-FILE

ON DESCENDING KEY HOURLY-RATE

ON ASCENDING KEY DEPT

ON ASCENDING KEY EMPLOYEE-ID

```
SD SORT-FILE...
01 SORT-REC.
   05 SORT-KEYS.
      10 DEPT          PIC XX.
      10 EMPLOYEE-ID   PIC X(4).
   05 NAME             PIC X(20).
   05 NUMBER-DEPENDENTS PIC S9      COMP-3.
   05 HOURLY-RATE       PIC S9(2)V99 COMP.
```

```
.....
SORT SORT-FILE
ON DESCENDING KEY HOURLY-RATE
ON ASCENDING KEY SORT-KEYS
```

١٢ - ٣٤ كيف يدمج الملفان التاليان :

File A: 10, 5, 18, 17, 23

File B: 20, 30, 40, 50

حيث إن الملف A غير مرتب .. فيجب أن يدمج الملفان باستخدام عبارة SORT :

```
SORT SORT-FILE
ON...
USING FILE-A FILE-B
OUTPUT PROCEDURE IS...
```

١٢ - ٣٥ حدد الخطأ فيما يلي :

```
PROCEDURE DIVISION.
OPEN INPUT FILE-A
FILE-B
MERGE MERGE-WORK-FILE
ON ASCENDING KEY MERGE-KEY
USING FILE-A
FILE-B
OUTPUT PROCEDURE IS PROCESS-RECORDS
```

يتطلب جزء USING أن يخلق ملف USING عند تنفيذ عبارة SORT ، أو عبارة MERGE (حيث تفتحه منفعة الترتيب أو الدمج تلقائياً) .

تمارين برهجة

١٢ - ٣٦ اكتب برنامج مطابقة لوكيل تجميع ديون:

- المدخلات : بطاقة تحتوى على رقم الحساب (الأعمدة ١ - ٦) ، الاسم (الأعمدة ١١ - ٢٠) ، المبلغ المدان به (الأعمدة ٣١ - ٣٧ ، مع تخصيص خانتين للكسر العشري) .

• المخرجات : تبدأ طباعة تقرير فى صفحة جديدة وله عنوان ، وعناوين أعمدة مناسبة ، مع طباعة رقم الحساب والا والمبلغ المدان به فى الصفحة . قم بعمل التخطيط الذى تراه مناسباً للتقرير ، ورتب الحسابات ترتيباً تنازلياً : طبقاً للمدان به العميل . اطبع فى النهاية إجمالى عدد العملاء ، وإجمالى المبالغ المدانين بها .

١٢ - ٣٧ اكتب برنامجاً لتساعد الجامعة فى تتبع الطلبة الذين لم يدفعوا ثمن المذكرات .

- المدخلات : بطاقات بها رقم تعريف الطالب (فى الصورة X(9)) واسمه (فى الصورة X(20)) ، والمبلغ المدان به الطالب (فى صورة 9(2)V9(2)) .

• المخرجات : طباعة تقرير له عنوان ، وعناوين أعمدة مناسبة . اطبع معلومات الطالب عبر الصفحة ، ويجب أن يسرد التقرير الطلبة : بترتيب الطالب صاحب أكبر دين أولاً ، وهكذا .. حتى يأتى الطالب صاحب أقل دين فى نهاية التقرير . اطبع فى نهاية التقرير إجمالى عدد الطلبة المدنين ، وإجمالى قيمة الدين ، ومتوسط قيمة الدين .

١٢ - ٣٨ اكتب برنامجاً ينتج تقرير الخطأ فى الأوامر ، وملف أوامر رئيسى .

• المدخلات :

Field Name	Characteristics	Position
Customer Number	9(6)	1-6
Customer Name	X(20)	11-30
Item Number	X(6)	31-36
Order Number	X(5)	38-42
Order Date	9(6)	44-49
Order Quantity	9(3)	51-53
Price	999V99	55-59

البطاقات لا تكون مرتبة ، ويعرف رقم العميل ، ورقم الأمر ، ورقم العنصر بأنها حقول حرفية عديدة .

• المخرجات :

١ - تقرير يحتوى على معلومات من هذه البطاقات تبين الأخطاء . وتأكد من صحة رقم العميل ، ورقم العنصر ، ورقم الأمر والتاريخ ، والكمية والسعر ، ومن أن السعر أكبر من صفر .

٢ - ملف أوامر رئيسى على قرص ، يجب أن يكون مرتباً طبقاً لرقم العميل ، مع الترتيب طبقاً لرقم العنصر لكل عميل . ويوجد سجل واحد فى هذا الملف لكل عميل . الشكل التخطيطى لهذا الملف ، كذلك كما يلى :

Field Name	Characteristics	Position
Customer Number	X(6)	1-6
Customer Name	X(20)	7-26
Number of Orders	9 (COMP-3)	27

يتبعه تاريخ لكل أمر :

Item Number	X(6)
Order Number	X(5)
Date	9(6)
Quantity	9(3)
Price	999V99

توجد أربعة أوامر على الأكثر للعميل الواحد ، وإجمالي طول السجل هو ١٢٧ بايت .

• **التشغيل :** التأكد من صحة سجلات المدخلات ، وطباعة تقرير بالبطاقات الخاطئة . يتم ترتيب البطاقات الصحيحة : طبقاً لرقم العنصر داخل رقم العميل ، مع استخدام سجلات مرتبة في إنتاج ملف رئيسي على قرص . إذا كانت هناك أكثر من أربعة سجلات للعميل الواحد .. فإنه لن يدخل إلا الأربعة سجلات الأولى في الملف الرئيسي ، هذا بالإضافة إلى طباعة رسائل خطأ للأوامر الزائدة مع إعمال بطاقتها .

١٢ - ٢٩ جدد بأوامر اليوم الملف الرئيسي الذي سبق إنتاجه في التمرين السابق . وقد تريد استخدام خوارزمي خط الاتزان ، الذي سبق التعرض له في الفصل الحادي عشر .

• المخرجات :

(١) ملف أوامر رئيسي نفس التخطيط الموجود في التمرين السابق ، ولكن به سجلات متغيرة الطول .
بالسجل أي طول مطلوب لاحتوائه على الأوامر التي يشملها . ويبلغ طول السجلات التي تحتوى على أمر واحد 52 رمزا ، ويبلغ طول السجلات التي تحتوى على 4 أوامر 127 رمزا . هذا الملف هو ملف قرص .
٢ - ملف عمليات جارية للأوامر نفس التخطيط الموجود في التمرين السابق ، وهو ملف بطاقات .

• المخرجات :

١ - ملف الأوامر الرئيسي المجدد .
٢ - مسجل عمليات جارية ، وتقرير مطبوع يبين التغيرات التي حدثت على الملف الرئيسي ، ويعرض رسائل الخطأ للسجلات التي لم تجدد في الملف الرئيسي .

• التشغيل :

يرتب الملف الرئيسي طبقاً لرقم العنصر داخل رقم العميل . رتب ملف الأوامر : بحيث يكون له نفس هذا الترتيب (سبق التأكد من صحة كل بطاقات الأوامر ، وتحتوى على حقول عديدة) في جزء التجديد .

إذا كان هناك أمر لعمل غير موجود فى الملف الرئيسى ، تطبع رسالة خطأ مع إهمال الأمر . إذا كان رقم العميل الموجود فى أحد الأوامر متفقاً مع نظيره فى الملف الرئيسى يحدث ما يلى :

إذا كان العنصر المطلوب موجوداً فعلاً فى الملف الرئيسى .. فإن هذا العنصر يتجدد بإضافة كمية الأمر إلى الكمية الموجودة فى الملف الرئيسى ، ويتغير تاريخ الأمر فى الملف الرئيسى ليصبح تاريخ اليوم . أما إذا لم يوجد العنصر المطلوب فى الملف الرئيسى . تطبع رسالة خطأ إذا لم يكن هناك مكان لإضافة العنصر إلى الملف الرئيسى .

أما إذا كان هناك مكان لذلك .. فيضاف الى الملف الرئيسى ، ومع إدخاله .. يظل الترتيب كما هو طبقاً لرقم العنصر .

تحذير :

تأكد من كتابة كل السجلات الرئيسية التى تقرأ ، حتى إذا لم تكن هناك أوامر من العملاء .

ملحق أ
كلمات كوبل المحجوزة
COBOL RESERVED WORDS

الكلمة المحجوزة	كوبل 1974 النمطى	كوبل CODASYL 80	كوبل IBM OS/VS	كوبل 1985 النمطى*
ACCEPT	x	-	x	x
ACCESS	x	-	x	x
ACTUAL	-	-	x	-
ADD	x	-	x	x
ADVANCING	x	-	x	x
AFTER	x	-	x	x
ALL	x	-	x	x
ALPHABET	x	-	x	x
ALPHABETIC	-	x	x	x
ALPHANUMERIC	x	-	-	x
ALPHANUMERIC - EDITED	-	x	x	x
ALSO	-	x	-	x
ALTER	x	-	-	x
ALTERNATE	x	-	x	x
AND	x	-	x	x
ANY	x	-	x	x
APPLY	-	x	x	x
ARE	-	-	-	x
AREA	x	-	x	-
AREAS	x	-	x	x
ASCENDING	x	-	x	x
ASSIGN	x	-	x	x
AT	x	-	x	x
AUTHOR	x	-	x	x

* إضافة من المترجم .

الكلمة المحجوزة	كويل 1974 التعطي	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 التعطي*
BASIS	-	X	-	-
BEFORE	X	-	X	X
BEGINNING	-	-	X	-
BINARY	-	X	-	X
BIT	-	X	-	-
BITS	-	X	-	-
BLANK	X	-	X	X
BLOCK	X	-	X	X
BOOLEAN	-	X	-	-
BOTTOM	X	-	X	X
BY	X	-	X	X
CALL	X	-	X	X
CANCEL	X	-	X	X
CBL	-	-	X	-
CD	X	-	X	X
CF	X	-	X	X
CH	X	-	X	X
CHANGED	-	-	X	-
CHARACTER	X	-	X	X
CHARACTERS	X	-	X	X
CLASS*	-	-	-	X
CLOCK - UNITS	X	-	-	-
CLOSE	X	-	X	X
COBOL	X	-	-	-
CODE	X	-	X	X
CODE - SET	X	-	X	X
COLLATING	X	-	X	X
COLUMN	X	-	X	X
COMMA	X	-	X	X
COMMIT	-	X	-	-
COMMON	-	X	-	X
COMMUNICATION	X	-	X	X
COMP	X	-	X	X
COMP - 1	-	-	X	-
COMP - 2	-	-	X	-
COMP - 3	-	-	X	-
COMP - 4	-	-	X	-
COMPUTATIONAL	X	-	X	X
COMPUTATIONAL - 1	-	-	X	-
COMPUTATIONAL - 2	-	-	X	-
COMPUTATIONAL - 3	-	-	X	-
COMPUTATIONAL - 4	-	-	X	-
COMPUTE	X	-	X	X

* إضافة من المترجم .

الكلمة المحجوزة	كويل 1974 النمطى	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 النمطى *
CONFIGURATION	X	-	X	X
CONNECT	-	X	-	-
CONSOLE	-	-	X	-
CONTAINS	X	-	X	X
CONTENT	-	X	-	X
CONTINUE	-	X	-	X
CONTROL	X	-	X	X
CONTROLS	X	-	X	X
CONVERTING	-	X	-	X
COPY	X	-	X	X
CORE - INDEX	-	-	X	-
CORR	X	-	X	X
CORRESPONDING	X	-	X	X
COUNT	X	-	X	X
CSP	-	-	X	-
CURRENCY	X	-	X	X
CURRENT	-	X	-	-
CURRENT - DATE	-	-	X	-
C 01	-	-	X	-
C 02	-	-	X	-
C 03	-	-	X	-
C 04	-	-	X	-
C 05	-	-	X	-
C 06	-	-	X	-
C 07	-	-	X	-
C 08	-	-	X	-
C 09	-	-	X	-
C 10	-	-	X	-
C 11	-	-	X	-
C 12	-	-	X	-
DATA	X	-	X	X
DATE	X	-	X	X
DATE - COMPILED	X	-	X	X
DATE - WRITTEN	X	-	X	X
DAY	X	-	X	X
DAY - OF - WEEK	-	-	X	X
DB	-	X	-	X
DB - ACCESS - CONTROL - KEY	-	X	-	-
DB - DATA - NAME	-	X	-	-
DB - EXCEPTION	-	X	-	-
DB - RECORD - NAME	-	X	-	-
DB - SET - NAME	-	X	-	-
DB - STATUS	-	X	-	-

* إضافة من المترجم

الكلمة المحجوزة	كويل 1974 النمطي	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 النمطي *
DE	x	-	x	x
DEBUG	-	-	x	x
DEBUG - CONTENTS	x	-	x	x
DEBUG - ITEM	x	-	x	x
DEBUG - LINE	x	-	x	x
DEBUG - NAME	x	-	x	x
DEBUG - SUB - 1	x	-	x	x
DEBUG - SUB - 2	x	-	x	x
DEBUG - SUB - 3	x	-	x	x
DEUGGING	x	-	x	x
DECIMAL - POINT	x	-	x	x
DECLARATIVES	x	-	x	x
DELETE	x	-	x	x
DELIMITED	x	-	x	x
DELIMITER	x	-	x	x
DEPENDING	x	-	x	x
DESCENDING	x	-	x	x
DESTINATION	x	-	x	x
DETAIL	x	-	x	x
DISABLE	x	-	x	-
DISCONNECT	-	-	-	-
DIS	-	-	x	x
DISPLAY	x	-	x	-
DISPLAY - n	-	x	-	-
DISPLAY - ST	-	-	x	x
DIVIDE	x	-	x	x
DIVISION	x	-	x	x
DOWN	x	-	x	x
DUPLICATE	-	x	-	-
DUPLICATES	x	-	x	x
DYNAMIC	x	-	x	x
EGI	x	-	x	x
EJECT	-	-	x	-
ELSE	x	-	x	x
EMI	x	-	x	x
EMPTY	-	x	-	-
ENABLE	x	-	x	x
END	x	-	x	x
END - ADD	-	x	-	x
END - CALL	-	x	-	x
END - COMPUTE	-	x	-	x
END - DELETE	-	x	-	x
END - DIVIDE	-	x	-	x

* إضافة من المترجم .

الكلمة المحجوزة	كويل 1974 النمطى	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 النمطى *
END - EVALUATE	-	X	-	X
END - IF	-	X	-	X
END - MULTIPLY	-	X	-	X
END - OF - PAGE	X	-	X	X
END - PERFORM	-	X	-	X
END - READ	-	X	-	X
END - RECEIVE	-	X	-	X
END - RETURN	-	X	-	X
END - REWRITE	-	X	-	X
END - SEARCH	-	X	-	X
END - START	-	X	-	X
END - STRING	-	X	-	X
END - SUBTRACT	-	X	-	X
END - UNSTRING	-	X	-	X
END - WRITE	-	X	-	-
ENDING	-	-	X	-
ENTER	X	-	X	-
ENTRY	-	-	X	X
ENVIRONMENT	X	-	X	X
EOP	X	-	X	X
EQUAL	X	-	X	-
EQUALS	-	X	-	-
ERASE	-	X	-	X
ERROR	X	-	X	X
ESI	X	-	X	X
EVALUATE	-	X	-	-
EVERY	X	-	X	-
EXAMINE	-	-	X	-
EXCEEDS	-	X	-	X
EXCEPTION	X	-	X	-
EXCLUSIVE	-	X	-	-
EXHIBIT	-	-	X	X
EXIT	X	-	X	-
EXOR	-	X	-	X
EXTEND	X	-	X	X
EXTERNAL	-	X	-	X
FALSE	-	X	-	X
FD	X	-	X	X
FILE	X	-	X	X
FILE - CONTROL	X	-	X	-
FILE - LIMIT	-	-	X	-
FILE - LIMITS	-	-	X	X
FILLER	X	-	X	X

* إضافة من المترجم

الكلمة المحجوزة	كويل 1974 التمطى	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 التمطى *
FINAL	x	-	x	-
FIND	-	x	-	-
FINISH	-	x	-	x
FIRST	x	-	x	x
FOOTING	x	-	x	x
FOR	x	-	x	-
FREE	-	x	-	x
FROM	x	-	x	-
FUNCTION	-	x	-	x
GENERATE	x	-	x	-
GET	-	x	-	x
GIVING	x	-	x	x
GLOBAL	-	x	-	x
GO	x	-	x	x
GREATER	x	-	x	x
GROUP	x	-	x	-
HEADING	x	-	x	x
HIGH - VALUE	x	-	x	x
HIGH - VALUES	x	-	x	x
1 - O	x	-	x	x
1 - O - CONTROL	x	-	x	-
ID	-	-	x	x
IDENTIFICATION	x	-	x	x
IF	x	-	x	x
IN	x	-	x	x
INDEX	x	-	x	-
INDEX -n	-	x	-	x
INDEXED	x	-	x	x
INDICATE	x	-	x	x
INITIAL	x	-	x	x
INITIALIZE	-	x	x	x
INITIATE	x	-	x	x
INPUT	x	-	x	x
INPUT - OUTPUT	x	-	x	-
INSERT	-	-	x	x
INSPECT	x	-	x	x
INSTALLATION	x	-	x	x
INTO	x	-	x	x
INVALID	x	-	x	x
IS	x	-	x	x

* إضافة من المترجم

الكلمة المحجوزة	كويل 1974 النمطى	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 النمطى *
JUST	X	-	X	X
JUSTIFIED	X	-	X	-
KEEP	-	X	-	X
KEY	X	-	X	X
LABEL	X	-	X	X
LAST	X	-	X	-
LD	-	X	-	X
LEADING	X	-	X	-
LEAVE	-	-	X	X
LEFT	X	-	X	X
LENGTH	X	-	X	X
LESS	X	-	X	X
LIMIT	X	-	X	X
LIMITS	X	-	X	X
LINAGE	X	-	X	X
LINAGE - COUNTER	X	-	X	X
LINE	X	-	X	X
LINE - COUNTER	X	-	X	X
LINES	X	-	X	X
LINKAGE	X	-	X	-
LOCALLY	-	X	-	X
LOCK	X	-	X	X
LOW - VALUE	X	-	X	X
LOW - VALUES	X	-	X	-
MEMBER	-	X	-	X
MEMORY	X	-	X	X
MERGE	X	-	X	X
MESSAGE	X	-	X	X
MODE	X	-	X	-
MODIFY	-	X	-	-
MODULES	X	-	X	X
MORE - LABELS	-	-	X	X
MOVE	X	-	X	X
MULTIPLE	X	-	X	-
MULTIPLY	X	-	X	X
NAMED	-	-	X	-
NATIVE	X	-	X	X
NEGATIVE	X	-	X	X
NEXT	X	-	X	-
NO	X	-	X	X
NOMINAL	-	X	X	-

* إضافة من المترجم

الكلمة المحجوزة	كويل 1974 التمطى	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 التمطى *
NOT	x	-	x	-
NOTE	-	-	x	x
NULL	-	x	-	x
NUMBER	x	-	x	x
NUMERIC	x	-	x	-
NUMERIC - EDITED	-	x	-	x
OBJECT - COMPUTER	x	-	x	x
OCCURS	x	-	x	x
OF	x	-	x	x
OFF	x	-	x	x
OMITTED	x	-	x	x
ON	x	-	x	x
OPEN	x	-	x	x
OPTIONAL	x	-	x	x
OR	x	-	x	x
ORDER	-	x	-	x
ORGANIZATION	x	-	x	-
OTHER	-	x	-	x
OTHERWISE	-	-	x	x
OUTPUT	x	-	x	-
OVERFLOW	x	-	x	x
OWNER	-	x	-	x
PACKED - DECIMAL	-	x	-	x
PADDING	-	x	-	x
PAGE	x	-	x	x
PAGE - COUNTER	x	-	x	-
PASSWORD	-	-	x	x
PERFORM	x	-	x	x
PF	x	-	x	x
PH	x	-	x	x
PIC	x	-	x	x
PICTURE	x	-	x	x
PLUS	x	-	x	x
POINTER	x	-	x	x
POSITION	x	-	x	-
POSITIONING	-	-	x	x
POSITIVE	x	-	x	x
PRINTING	x	-	-	-
PRIOR	-	x	-	x
PROCEDURE	x	-	x	x
PROCEDURES	x	-	x	x
PROCEED	x	-	x	-

* إضافة من المترجم

الكلمة المحجوزة	كويل 1974 النعطي	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 النعطي *
PROCESSING	-	-	X	X
PROGRAM	X	-	X	X
PROGRAM - ID	X	-	X	X
PROTECTED	-	X	-	X
PURGE	-	X	-	X
QUEUE	X	-	X	X
QUOTE	X	-	X	X
QUOTES	X	-	X	X
RANDOM	X	-	X	-
RD	X	-	X	-
READ	X	-	X	-
READY	-	-	X	X
REALM	-	X	-	-
REALMS	-	X	-	X
RECEIVE	X	-	X	-
RECONNECT	-	X	-	-
RECORD	X	-	X	X
RECORD - NAME	-	X	-	X
RECORD - OVERFLOW	-	-	X	X
RECORDS	X	-	X	X
REDEFINES	X	-	X	X
REEL	X	-	X	X
REFERENCE	-	X	-	X
REFERENCES	X	-	X	-
RELATIVE	X	-	X	X
RELEASE	X	-	X	-
RELOAD	-	-	X	X
REMAINDER	X	-	X	X
REMARKS	-	-	X	-
REMOVAL	X	-	X	-
RENAMES	X	-	X	X
REORG - CRITERIA	-	-	X	X
REPEATED	-	X	-	X
REPLACE	-	X	-	X
REPLACING	X	-	X	X
REPORT	X	-	X	-
REPORTING	X	-	X	-
REPORTS	X	-	X	X
REREAD	-	-	X	X
RERUN	X	-	X	-
RESERVE	X	-	X	-
RESET	X	-	X	X
RETAINING	-	X	-	-

* إضافة من المترجم

الكلمة المحجوزة	كويل 1974 النمطى	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 النمطى*
RETRIEVAL	-	X	-	-
RETURN	X	-	X	X
RETURN - CODE	-	-	X	X
REVERSED	X	-	X	X
REWIND	X	-	X	X
REWRITE	X	-	X	X
RF	X	-	X	-
RH	X	-	X	X
RIGHT	X	-	X	X
ROLLBACK	-	X	-	X
ROUNDED	-	-	X	X
RUN	X	-	X	X
SAME	X	-	X	X
SD	X	-	X	X
SEARCH	X	-	X	-
SECTION	X	-	X	X
SECURITY	X	-	X	X
SEEK	-	-	X	X
SEGMENT	X	-	X	-
SEGMENT - LIMIT	X	-	X	X
SELECT	X	-	X	X
SELECTIVE	-	-	X	X
SEND	X	-	X	X
SENTENCE	X	-	X	X
SEPARATE	X	-	X	X
SEQUENCE	X	-	X	-
SEQUENTIAL	X	-	X	X
SET	X	-	X	X
SETS	-	X	-	-
SIGN	X	-	X	-
SIZE	X	-	X	-
SKIP-1	-	-	X	X
SKIP-2	-	-	X	-
SKIP-3	-	-	X	-
SORT	X	-	X	X
SORT-CODE - SIZE	-	-	X	-
SORT-FILE - SIZE	-	-	X	-
SORT-MERGE	X	-	X	-
SORT-MESSAGE	-	-	X	X
SORT-MODE - SIZE	-	-	X	X
SORT-RETURN	-	-	X	X
SOURCE	X	-	X	X
SOURCE - COMPUTER	X	-	X	X

* إضافة من المترجم .

الكلمة المحجوزة	كويل 1974 التمطى	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 التمطى*
SPACE	X	-	X	X
SPACES	X	-	X	X
SPECIAL - NAMES	X	-	X	X
STANDARD	X	-	X	X
STANDARD-1	X	-	X	X
STANDARD-2	-	X	-	X
START	X	-	X	-
STATUS	X	-	X	X
STOP	X	-	X	X
STORE	-	-	X	X
STRING	X	-	X	X
SUB-QUEUE-1	X	-	X	-
SUB-QUEUE-2	X	-	X	X
SUB-QUEUE-3	X	-	X	X
SUB - SCHEMA	-	X	-	X
SUBTRACT	X	-	X	X
SUM	X	-	X	X
SUPPRESS	X	-	X	X
SYMBOLIC	X	-	X	-
SYNC	X	-	X	-
SYNCHRONIZED	X	-	X	-
SYSIN	-	-	X	-
SYSOUT	-	-	X	-
SYSPUNCH	-	-	X	X
S 01	-	-	X	-
S 0 2	-	-	X	-
TABLE	X	-	X	X
TALLY	-	-	X	-
TALLYING	X	-	X	X
TAPE	X	-	X	X
TENANT	-	X	-	X
TERMINAL	X	-	X	X
TERMINATE	X	-	X	X
TEST	-	X	-	X
TEXT	X	-	X	X
THAN	X	-	X	X
THEN	-	-	X	-
THROUGH	X	-	X	X
THRU	X	-	X	X
TIME	X	-	X	-
TIME - OF - DAY	-	-	X	X
TIMES	X	-	X	X
TO	X	-	X	X

* إضافة من المترجم .

الكلمة المحجوزة	كويل 1974 التمطى	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 التمطى *
TOP	x	-	x	-
TOTALED	-	-	x	-
TOTALING	-	-	x	-
TRACE	-	-	x	x
TRACK - AREA	-	-	x	-
TRACK - LIMIT	-	-	x	x
TRACKS	-	-	x	x
TRAILING	x	-	x	x
TRANSFORM	-	-	x	-
TRUE	-	x	-	x
TYPE	x	-	x	-
UNEQUAL	-	x	-	x
UNIT	x	-	x	x
UNSTRING	x	-	x	-
UNTIL	x	-	x	x
UP	x	-	x	-
UPDATE	-	x	-	-
UPON	x	-	x	-
UPSI - 0	-	-	x	-
UPSI - 1	-	-	x	-
UPSI - 2	-	-	x	-
UPSI - 3	-	-	x	-
UPSI - 4	-	-	x	-
UPSI - 5	-	-	x	x
UPSI - 6	-	-	x	-
UPSI - 7	-	-	x	x
USAGE	x	-	x	x
USAGE - MODE	-	x	-	x
USE	x	-	x	x
USING	x	-	x	x
VALUE	x	-	x	x
VALUES	x	-	x	x
VARYING	x	-	x	x
WHEN	x	-	x	-
WHEN - COMPILED	-	-	x	x
WITH	x	-	x	-
WITHIN	-	x	-	x
WORDS	x	-	x	x
WORKING - STORAGE	x	-	x	x
WRITE	x	-	x	-
WRITE - ONLY	-	-	x	x

* إضافة من المترجم .

الكلمة المحجوزة	كويل 1974 النعطي *	كويل CODASYL 80	كويل IBM OS/VS	كويل 1985 النعطي *
ZERO	X	-	X	X
ZEROES	X	-	X	X
ZEROS	X	-	X	X
ALPHABETIC - LOWR *	-	-	-	X
ALPHABETIC - UPPER *	-	-	-	X
+	X	-	X	X
-	X	-	X	X
*	X	-	X	X
/	X	-	X	X
**	X	-	X	X
<	X	-	X	X
>	X	-	X	X
=	X	-	X	X
> = *	-	-	-	X
< = *	-	-	-	X

* إضافة من المترجم .

ملحق ب
تسلسل التتابع
Collating Sequences

الترتيب التصاعدي لنظام EBCDIC

التمثيل العشري	تشكيل البت	الرمز	المعنى
74	01001010	#	- علامة الست
75	01001011	.	- نقطة أو علامة عشرية
76	01001100	<	- إشارة أقل من
77	01001101	(- قوس أيسر
78	01001110	+	- إشارة موجب
79	01001111		- عمود رأسى ، أو منطقية
80	01010000	&	- علامة إضافة
90	01011010	!	- علامة تعجب
91	01011011	\$	- علامة دولار
92	01011100	*	- نجمة
93	01011101)	- قوس أيمن
94	01011110	:	- فاصلة منقولة
95	01011111	~	- NOT منطقية
96	01100000	-	- إشارة سالب أو شرطة
97	01100001	/	- شرطة مائلة
107	01101011	°	- فاصلة

التمثيل العشري	تشكيل البت	الرمز	المعنى
108	01101100	%	- علامة النسبة المئوية
109	01101101	-	- شرطة تحت الحرف
110	01101110	>	- إشارة أكبر من
111	01101111	?	- علامة استفهام
122	01111100	:	- نقطتان علويتان
123	01111101	#	- إشارة الرقم
124	01111110	@	- إشارة عند
125	01111111	,	- علامة تنصيص فردية
126	01111110	=	- علامة تساوى
127	01111111	~	- علامة تنصيص مزدوجة
129	10000001	a	
130	10000010	b	
131	10000011	c	
132	10000100	d	
133	10000101	e	
134	10000110	f	
135	10000111	g	
136	10001000	h	
137	10001001	i	
145	10010001	j	
146	10010010	k	
147	10010011	l	
148	10010100	m	
149	10010101	n	
150	10010110	o	
151	10010111	p	
152	10011000	q	

التمثيل العشري	تشكيل البت	الرمز	المعنى
153	10011001	r	
162	10100010	s	
163	10100011	t	
164	10100100	u	
165	10100101	v	
166	10100110	w	
167	10100111	x	
168	10101000	y	
169	10101001	z	
193	11000001	A	
194	11000010	B	
195	11000011	C	
196	11000100	D	
197	11000101	E	
198	11000110	F	
199	11000111	G	
200	11001000	H	
201	11001001	I	
209	11010001	J	
210	11010010	K	
211	11010011	L	
212	11010100	M	
213	11010101	N	
214	11010110	O	
215	11010111	P	
216	11011000	Q	

التمثيل العشري	تشكيل البت	الرمز	المعنى
217	11011001	R	
226	11100010	S	
227	11100011	T	
228	11100100	U	
229	11100101	V	
230	11100110	W	
231	11100111	X	
232	11101000	Y	
233	11101001	Z	
240	11110000	0	
241	11110001	1	
242	11110010	2	
243	11110011	3	
244	11110100	4	
245	11110101	5	
246	11110110	6	
247	11110111	7	
248	11111000	8	
249	11111001	9	

الترتيب التصاعدي لنظام ASCII

التمثيل العشري	تشكيل البت	الرمز	المعنى
0	00000000		- لا شيء (صفر)
32	00100000	SP	- فراغ
33	00100001		- OR منطقية
34	00100010	"	- علامة تنصيص مزدوجة
35	00100011	#	- إشارة رقم
36	00100100	\$	- علامة دولار
37	00100101	%	- نسبة مئوية
38	00100110	&	- علامة إضافة
39	00100111	'	- علامة تنصيص فردية
40	00101000	(- قوس مفتوح
41	00101001)	- قوس مغلق
42	00101010	*	- نجمة
43	00101011	+	- زائد
44	00101100	,	- فاصلة
45	00101101	-	- ناقص أو شرطة
46	00101110	.	- نقطة أو علامة عشرية
47	00101111	/	- شرطة مائلة للخلف
48	00110000	0	
49	00110001	1	
50	00110010	2	
51	00110011	3	
52	00110100	4	
53	00110101	5	
54	00110110	6	
55	00110111	7	
56	00111000	8	
57	00111001	9	

التمثيل العشري	تشكيل البت	الرمز	المعنى
58	00111010	:	- نقطتان علويتان
59	00111011	:	- فاصلة متقولة
60	00111100	<	- أقل من
61	00111101	=	- يساوي
62	00111110	>	- أكبر من
63	00111111	?	- علامة استفهام
64	01000000	@	- إشارة عند
65	01000001	A	
66	01000010	B	
67	01000011	C	
68	01000100	D	
69	01000101	E	
70	01000110	F	
71	01000111	G	
72	01001000	H	
73	01001001	I	
74	01001010	J	
75	01001011	K	
76	01001100	L	
77	01001101	M	
78	01001110	N	
79	01001111	O	
80	01010000	P	
81	01010001	Q	
82	01010010	R	
83	01010011	S	
84	01010100	T	

التمثيل العشري	تشكيل البت	الرمز	المعنى
85	01010101	U	
86	01010110	V	
87	01010111	W	
88	01011000	X	
89	01011001	Y	
90	01011010	Z	
91	01011011	[قوس مربع مفتوح
92	01011100	\	- شرطة مائلة عكسية
93	01011101]	- قوس مربع مغلق
94	01011110	^	- منطقية
95	01011111	-	- شرطة تحت الحرف
96	01100000	`	grave accent
97	01100001	a	
98	01100010	b	
99	01100011	c	
100	01100100	d	
101	01100101	e	
102	01100110	f	
103	01100111	g	
104	01101000	h	
105	01101001	i	
106	01101010	j	
107	01101011	k	
108	01101100	l	
109	01101101	m	

التمثيل العشري	تشكيل البت	الرمز	المعنى
110	01101110	n	
111	01101111	o	
112	01110000	p	
113	01110001	q	
114	01110010	r	
115	01110011	s	
116	01110100	t	
117	01110101	u	
118	01110110	v	
119	01110111	w	
120	01111000	x	
121	01111001	y	
122	01111010	z	
123	01111011	{	- قوس [مفتوح
124	01111100	:	- خط علوي
125	01111101	}	- قوس مغلق]
126	01111110	~	tilde

ملحق ج

اعتبارات كوبل الثمانينيات

Cobol' 80 Considerations

تعمل لجان ANSI, CODASYL حديثاً (طبقاً لما جاء في الكتاب) على صيغة جديدة لكوبل للثمانينيات الميلادية. وكأول محاولة أذيعت ، عام 1981 م ، هي كوبل 80 النمطى ، الذى يشمل تغييرات تجعل بعض المبرمجين بصيغة كوبل 1974 النمطية غير متوافقين مع كوبل 80 . وحالياً تجرى مراجعة أكثر للنمطية الجديدة ، مع وجود نقد بناء وحوار حولها . وفيما يلى بعض التغييرات الأكثر أهمية التى يمكن أن يتفق عليها :

- ١ - القيد على استخدام 3 دلائل (أو فهراس) ، و 3 مستويات لأجزاء OCCURS متداخلة أزيل (الى 48) .
- ٢ - يمكن تعريف السلاسل الجزئية لعناصر البيانات فى عبارات جزء الإجراءات : يحدد 7 بايت للعنصر DATA - ITEM الذى يبدأ بالبايت الثالث .
- ٣ - يمكن أن تكون للعنصر DEPENDING ON فى جزء OCCURS ... DEPENDING ON القيمة صفر (محدداً محتويات صفرية للجدول) .
- ٤ - يمكن لعنصر البيانات الذى يعرف عنصر بيانات آخر أن يكون أقصر منه .
- ٥ - تنتج ACCEPT ... FROM DAY - OF - WEEK رقماً مناسباً يقع بين 1 و 7 .
- ٦ - يسمح الآن بكلمة TO مع GIVING فى عملية الجمع .
ADD ITEM - A TO ITEM - B GIVING ITEM - C.
- ٧ - يمكن استخدام عبارة MOVE لنقل عنصر منقح إلى عنصر عددي غير منقح ، ولا يسري التنقيح أثناء العملية .
- ٨ - يسمح الآن بسجلات متغيرة الطول مع WRITE ... FROM, RETURN ... INTO, READ ... INTO, RELEASE ... FROM .
- ٩ - يتغير وصف محتويات المسجل الخاص DEBUG - ITEM ؛ تبعاً لتأثير ذلك على USE FOR DEBUGGING .
DECLARATIVES

- ١٠ - عندما يكون عنصر DEPENDING ON جزءاً من سجل متغير الطول ، ويكون حقلاً مستقبلاً ، تستخدم أقصى قيمة ممكنة لعنصر DEPENDING ON (بدلا من قيمته الحالية) في تحديد طول الحقل المستقبلي .
- ١١ - هناك شرطا فئة CLASS جديداً ، هما :
- ALPHABETIC - LOWER, ALPHABETIC - UPPER . يغطي شرط ALPHABETIC الحالي كلأ من الحروف العليا (الحروف الكبيرة) ، والحروف الدنيا (الحروف الصغيرة) .
- ١٢ - حذفت عبارة ALTER من اللغة .
- ١٣ - ينتهي تقويم التعبيرات الشرطية المركبة بمجرد تحديد القيمة الحقيقية النهائية للتعبير . وعلى هذا .. إذا كان A خطأ في "IF A AND B AND C..." .. فلن يقوم B أو C ؛ لأنه أصبح معروفاً على الفور أن التعبير خطأ .
- ١٤ - اتسع تعريف PERFORM للسماح بتحديد العبارات التي تنفذ في سطر على التوالي :

```
PERFORM UNTIL FLAG-IS-SET
  MOVE EMPLOYEE-NAME TO DETAIL-LINE-NAME
  ADD 1 TO NUMBER-PRINTED
  WRITE DETAIL-LINE ...
  PERFORM SOME-CALCULATION
  ADD EMPLOYEE-AMOUNT TO TOTAL-AMOUNT
END-PERFORM
```

قواعد تنفيذ العبارات المحددة هي نفسها كما لو كانت العبارات موجودة بمقطع مستقل بها (وإيكن A - PARA مثلا) وكان يستخدم مايلي :

```
PERFORM PARA-A UNTIL FLAG-IS-SET
```

لاحظ أنه عندما تحدد العبارات المراد تنفيذها في سطر على التوالي: (١) لا يوضع اسم مقطع أو قسم في عبارة PER-FORM (٢) ويكون مطلوب عبارة جديد END - PERFORM ؛ لتحديد نهاية مجموعة العناصر المشمولة في عبارة PER-FORM .

١٥ - اتسعت PERFORM ... UNTIL لتسمح بمواصفات تبين ضرورة اختبار شرط UNTIL قبل أو بعد تنفيذ العبارات أو المقاطع المحددة :

```
PERFORM SAMPLE-PARAGRAPH
  WITH TEST AFTER
  UNTIL WE-ARE-DONE
```

أو

```

PERFORM WITH TEST BEFORE
  UNTIL TEST-IS-SATISFIED
    ADD ...
    MOVE ...
    READ ...
END-PERFORM

```

باستخدام TEST - AFTER .. تنفذ العبارات المحددة مرة واحدة على الأقل، حتى إذا كان الشرط متحققا في البداية (DO UNTIL) ، وباستخدام TEST - BEFORE (الحالة التقليدية) .. لا تنفذ العبارات المحددة عندما يكون الشرط متحققا في البداية .

١٦ - تنتهي عبارة IF بعبارة جديدة هي IF - END بدلا من نهايتها باستخدام النقطة .

```

IF A EQUAL B
  ADD ...
  MOVE ...
END-IF

```

أو

```

IF X NOT EQUAL Y
  MOVE ...
  PERFORM ...
ELSE
  ADD ...
  READ ...
END-IF

```

١٧ - أضيفت عبارة جديدة EVALUATE إلى الكويل لتنفيذ هيكل الحالة . ونوضح ذلك بإعادة كتابة مثال (٧ - ٢٦) على النحو التالي:

```

EVALUATE TRANSACTION-CODE
  WHEN 1 PERFORM CREATE-NEW-MASTER-RECORD
  WHEN 2 PERFORM DELETE-MASTER-RECORD
  WHEN 3 PERFORM CHANGE-EXISTING-MASTER-RECORD
  WHEN OTHER
    PERFORM INVALID-TRANSACTION-CODE-ROUTINE
END-EVALUATE

```

لاحظ جزء اصطلياد الخطأ WHEN OTHER ، والإنتهاء الصريح END-EVALUATE .

كلمة من المترجم عن الاختلافات الرئيسية بين كويل 1985 النمطى ، وكويل 1974 النمطى

لقد ظهرت فعلا صيغة الكويل القياسية المعروفة بكويل ١٩٨٥ النمطى ، وفيما يلي الاختلافات الرئيسية بينها وبين كويل ١٩٧٤ النمطى ، هذا .. مع ملاحظة ملحق ١ ، وأن هناك كلمات محجوزة أضيفت إلى صيغة كويل ١٩٨٥ النمطية ، مع حذف بعض كلمات من صيغة كويل ١٩٧٤ النمطية منها .

١ - أصبح من الممكن استخدام كلمتي TO , GIVING في نفس عبارة الجمع .

٢ - أصبح في الإمكان استخدام مؤشرات لإنهاء العمليات الحسابية ، وهي:

END - ADD
END - SUBTRACT
END - DEVIDE
END - MULTIPLY
END - COMPUTE

كما أصبح في الإمكان استخدام مؤشرات لإنهاء عبارات أخرى على النحو التالي :

END - IF
END - EVALUATE
END - PERFORM
END - READ
END - SEARCH
END - WRITE

واستخدام هذه المؤشرات يجعل العبارات الشرطية أكثر ترتيباً مما يسهل من البرمجة المرتبة .

٣ - أصبح من الممكن استخدام كلمة THEN عند كتابة الشروط ؛ مما يجعل الكويل متفقاً مع هيكل IF - THEN - ELSE بالنسبة لهيكل الاختيار المرتب .

٤ - أصبح من الممكن استخدام PERFORM ، وكتابة العبارات المراد تنفيذها في سطر على التوالي . (الملاحظة رقم ١٤ أعلاه)

٥ - أصبح بالإمكان استخدام جزء TEST AFTER في عبارة PERFORM .

٦ - أصبح من الممكن وضع قيم ابتدائية لمجموعة عناصر باستخدام فعل INITIALIZE .

٧ - أصبح في الإمكان استخدام عدد من المستويات يصل إلى 7 مع OCCURS .

٨ - يمكن لعنصر OCCURS أن يحتوى على محتويات ابتدائية باستخدام جزء VALUE ، وهذا يلغى الحاجة الى REDE- FINE الجول أو المنظومة .

٩ - أدخل فعلاً جديداً EVALUATE وذلك لتنفيذ هيكل الحالة بالكويل .

١٠ - أصبح في الإمكان نقل عناصر منقحة إلى عناصر عديدة .

١١ - أصبح في الإمكان استخدام كلمة الكويل المحجوزة DAY - OF - WEEK التي تشغل محتوياتها خانة واحدة ؛ للدلالة على رقم اليوم في الأسبوع والذي يتراوح من 1 إلى 7 .

١٢ - أصبح النقل النسبي ممكناً من حقل لآخر ؛ فيمكن نقل جزء من حقل كبير إلى حقل صغير .

١٣ - أصبح من الممكن للتوابت غير العددية أن تشغل حتى 160 رمز ، بينما كان الحد الأعلى في كويل ١٩٧٤ النمطى هو 120 رمز فقط .

١٤ - لا تحتاج عبارة EXIT إلى أن تكون هي العبارة الوحيدة في المقطع الذى يحتوئها ، بينما - من الممنوع في كويل ١٩٧٤ النمطى كتابة أى شيء في المقطع المكتوب فيه EXIT باستثنائها هي بالطبع .

- ١٥ - يمكن أن تكون أسماء الإجراءات المستخدمة مع عبارة SORT أسماء مقاطع أو أسماء أقسام ، بينما يشترط فيها أن تكون أسماء أقسام فى كويل ١٩٧٤ النمطى.
- ١٦ - أصبح قسم التشكيل اختياريًا ، فى الواقع .. أصبح جزء التشكيل اختياريًا.
- ١٧ - أصبحت RECORD CONTAINS, BLOCK CONTAINS اختياريًا .
- ١٨ - أضيف المؤثران الحسابيان
(> =) IS GREATER THAN OR EQUAL , (< =) IS LESS THAN OR EQUAL ,
- ١٩ - أصبح من الممكن استخدام نفى عند حدوث خطأ فى الحجم ، ونهاية الملف ، والمفتاح غير الصحيح ؛ أى أصبح مسموحاً باستخدام ، و NOT AT END , NOT ON SIZE ERROR , NOT INVALID KEY .
- ٢٠ - أضيف الجزء WITH NO ADVANCING إلى عبارة DISPLAY ؛ لجعل التداخل بين مشغل الكمبيوتر والكمبيوتر أكثر صداقة .
- ٢١ - أصبح من الممكن استخدام الحروف الأبجدية الصغيرة فى الثوابت الحرفية عديدة . وتعتبر كائنها حرفية ، وتجتاح اختبار
. ALPHABETIC

المصطلحات العلمية (إنجليزي عربي)

A

A (picture character)	A (رمز صورة)
A margin	المنطقة
Abbreviaation	أختصار
Abbreviation of compound conditions	اختصار شروط مركبة
ABEND	نهاية غير طبيعية
Absolute value	قيمة مطلقة
ACCEPT	عبارة اقبل
Access arm	ذراع اتصال
ACCESS IS	الاتصال يكون
Access modes	حالات الاتصال
Access time	وقت الاتصال
Accumulators	مركبات
Action Stub	جزء أجزاء وهي
ADD	تجمع
Adding an entry	إضافة محتوى
Address	عنوان
Address calculations:	حسابات عنوان
AFTER	بعد
Advancing mnemonic - name	مع تقديم اسم خاص
ADVANCING PAGE	مع تقديم صفحة
AFTER ADVANCING	بعد تقديم
Algorithm	خوارزمية
ALL literal	كل ثوابت
ALPHABETIC	أبجدي

Alphabetic data	بيانات حرفية
Alphameric (alphanumeric) data	بيانات حرفية عددية (غير عددية)
ALU (arithmetic-logic unit)	وحدة حسابات ومنطق
AND	و
Applications program	برنامج تطبيق
Arithmetic, efficiency of	حساب ، كفاءة
Arithmetic- operators	مؤثرات حسابية
Arrays,	منظومات
ASCENDING KEY	مفتاح تصاعدي
ASCENDING/DESCENDING KEY	مفتاح تصاعدي أو تنازلي
Ascending sort	ترتيب تصاعدي
ASCII	الشفرة الأمريكية القياسية لتبادل المعلومات
ASSIGN TO	محدد له
Assumed	مفترضة
Assumed decimal point	علامة عشرية مفترضة
Asterisk (editing character)	نجمة (رمز تنقيح)
AT END	عند النهاية
AT END-OF-PAGE	عند نهاية الصفحة
AT EOP	عند نهاية الصفحة
AUTHOR	مؤلف
Automatic page overflow	سريان زائد تلقائي للصفحة
Auxiliary storage	تخزين مساعد

B

B (picture character)	B (رمز صورة)
B margin	المنطقة ب
Backup	احتياطي
Balanced-line sequential file update	تجديد ملف توازن تتابعي
Batch processing	تشغيل دفعات
Binary digit,	رقم ثنائي
Binary operators,	مؤثرات ثنائية
Binary search,	بحث ثنائي
Binary-twos-complement data	بيانات مكمل ازواج ثنائية
Bit	بت

Blank lines,	أسطر فارغة
BLANK WHEN ZERO	فراغات بدلا من الأصفار
Block:	مجموعة
BLOCK CONTAINS,	المجموعة تحتوى
BLOCK CONTAINS 0 CHARACTERS	المجموعة تحتوى صفرا من الرموز
Block descriptor word	كلمة واصف مجموعة
Blocking	تجميع
Blocking factor	معامل التجميع
Bottom margin	الهامش السفلى
Braces,	أقواس
Brackets	أقواس
Branch instruction	أمر تفريع
Bubble	فقاعة
Bubble Sort	ترتيب الفقاعة
Buffer	ذاكرة احتياطية
Byte	بايت

C

Calculating Sort record fields	حساب حقول سجل ترتيب
Card punch	مكتب بطاقات
Card reader	قارئ بطاقات
Carriage control C01,C02	تحكم العربى
Case structure	هيكل الحالة
COBOL	كوبل
CHANGED	تغيير
CHANGED NAMES	أسماء متغيرة
Check protection	حماية الشيكات
Class condition	شرط فئة
CLOSE	أغلق
COBOL	كوبل
Coding form	صيغة كتابة شفرة
Coding - testing plan	خطة اختبار الشفرة
Coding standards	نمطيات كتابة شفرة
Cohesion of modules	تماسك الأجزاء
Construction Of sort records	اعداد سجلات ترتيب

Collating sequence,	تسلسل التتابع
COLLATING SEQUENCE IS	تسلسل التتابع هو
Column	عمود
Comma (editing character)	فاصلة (رمز تنقيح)
Command language	لغة أوامر
Commands	أوامر
Comments	تعليقات
Completion of documetation	اتمام التوثيق
COMP	حسابي
COMP - 3	حسابي - ٣
Compared to subscript	مقارنة بدليل
COMP data	بيانات مضغوطة
COMP - 3 data	بيانات مضغوطة - ٣
COMPUTATIONAL	حسابي
COMPUTATIONAL - 3	حسابي - ٣
Comparison:	مقارنه
Compilation (diagram)	ترجمة (رسم)
Compiler	مترجم
Compound conditions	شروط مركبة
COMPUTE	احسب
Computer system	نظام كمبيوتر
Condition,	شرط
Condition-names	أسماء شرطية
Condition stub	جزء شرطي وهمي
Conditional error	خطأ شرطي
Conditional statements	عبارات شرطية
CONFIGURATION SECTION	قسم التشغيل
Connector symbol	رمز الواصل
CONSOLE	شاشة مرئية
Console terminal	نهاية طرفية بها شاشة
Continuation rules	قواعد مستمرة
Control break	تحكم متقطع
Control cards	بطاقات تحكم
Control fields	حقول تحكم
Control totals	إجماليات تحكم

Control unit (CU)	وحدة تحكم
Conversions	تحويلات
Converting input to subscripts	تحويل مدخلات إلى دلائل
Counters	عدادات
Counter of active entries	عداد المحتويات النشيطة
Coupling	تقارن
CR (picture character)	CR (رمز صورة)
CRT	أنبوبة أشعة الكاثود
CRT (terminals)	نهايات طرفية بأنبوب أشعة الكاثود
Cylinder	اسطوانة

D

Data Codes	شفرات (رموز) بيانات
Data division	جزء البيانات
Data Records	سجلات تاريخ
Date - Compiled	تاريخ الترجمة
Date fields	حقول تواريخ
Date - Written	تاريخ الكتابة
DAY	يوم
DB (picture character)	DP (رمز صورة)
Debug - Item	عنصر تصحيح
Debugging	تصحيح
Debugging Declarative	توضيحات تصحيح
Debugging lines	أسطر تصحيح
Decimal Point	علامة عشرية
Decision symbol	رمز قرار
Decision table	جدول قرارات
decoding input fields	فك شفرة حقول مدخلات
diagram	رسم
Decision trees	أشجار قرارات
DECLARATIVES	توضيحات
definition of	تعريف
Decoding input fields	فك شفرة حقول مدخلات
deleting an entry	حذف محتوى
Deletion code	رمز الحذف

DESCENDING	تنازلى
DESCENDING KEY	مفتاح تنازلى
Descending sort	ترتيب تنازلى
detailed desing	تصميم تفصيلى
Designing program logic (main loop)	تصميم منطقة البرنامج (ليرة رئيسية)
Diagnostic error messages	رسائل أخطاء تشخيصية
Direct-access device	وحدة اتصال مباشر
Directory	دليل
Disk:	قرص
DISPLAY	أعرض
diagram	رسم
Disk drive	مشغل أقراص
Disk pack	مجموعة أقراص
Displacement	إزاحة
DISPLAY	اعرض
DISPLAY data	اعرض بيانات
DISPLAYUSAGE	الاستخدام للعرض
DIVIDE	اقسم
DIVISION	قسمة
Division beader	عنوان قسمة
Do while	افعل أثناء
Dollar sign	علامة دولار (رمز تنقيح)
DOWY BY	لأسفل بـ
Dump	نفايا
dumb	غبية
Duplication Factor	معامل ازواج
Dynamic memory dump (DISPLAY)	نفايا ذاكرة ديناميكية (اعرض)

E

EBCDIC	الشفرة الثنائية المؤسسية للتبادل البشرى
Edit program	برنامج تنقيح
Editing character	رمز تنقيح
Editing	تنقيح
EJECT	يلفظ
Elementary item	عنصر فردى
ELSE	وإلا

ELSE NEXT SENTENCE	وإلا فالجمله التاليه
End of file	نهاية الملف
ENVIRONMENT DIVISION	جزء الاوساط
EQUAL	يساوى
Errors:	أخطاء
I-O	مدخلات ومخرجات
Evaluation	تقييم
EVALUATE	قويم
Example	مثال
Exception conditions	شروط استثناءات
Execution of Stored program	تنفيذ برنامج مخزن
Executive module	جزء تنفيذ
EXHIBIT	عرض
External name	اسم خارجى
EXTEND	موسع

F

Fatal logic	منطقى جسيم
Fatal logic errors	أخطاء منطقية جسيمة
FD entry	محتوى وصف الملف
Field	حقل
Figurative constant	ثابت استعارى
File	ملف
FILE-CONTROL	التحكم فى الملفات
File creation	إنتاج ملف
File description entry	محتوى وصف الملف
File labels	عناوين الملفات
File maintenance	صيانة الملف
File merging	دمج الملفات
File organization	تنظيم الملف
File retrieval	استرجاع ملف
FILE SECTION	قسم الملفات
File sorting	ترتيب الملف
Fixed - length	ثابتة الطول
Fixed	ثابت

FILE STATUS	حالة الملف
File updating	تجديد الملف
FILLER	ماليء
First-record processing	تشغيل أول سجل
First-record switch	مفتاح أول سجل
Fixed head-per-track disk	رأس ثابت لكل مسار
Flags,	إشارات
Floppy disk	قرص مرن
Floating	متحركة
Flowchart	خريطة مسار
Footing area,	منطقة نهايات
For sort file	ملف الترتيب
From input data	من بيانات المدخلات
Functional cohesion	تماسك وظيفي

G

Garbage	نفايا
General design	تصميم عام
GIVING	مفطيا
GO TO	اذهب إلى
Grandparent, parent, child back	نسخ احتياطية للجد والاب
GREATER THAN	أكبر من
Group	مجموعة
Group item	مجموعة عناصر
Groupitem for key	مجموعة عناصر المفتاح
Group MOVE	نقل جماعي

H

Hard copy terminals	نهايات طرفية لإنتاج نسخ دائمة
Hard copy	نسخة دائمة
Hard disk	قرص ثابت
Hardware	نظم مكونات
Header	عنوان
Header label	عنوان امامي
Hierarchy chart	خريطة هرمية

HIGHVALUES

أعلى قيمة في تسلسل التتابع

Higher-level language

لغة مرتفعة المستوى

HIPO chart

خريطة هيبو

Horizontal numbering

ترقيم أفقي

I

I-O

مدخلات ومخرجات

I-O errors

أخطاء مدخلات ومخرجات

IAR (Instruction Address Register)

مسجل عنوان إحدى التعليمات

IDENTIFICATION DIVISION

جزء التعريف

IF

إذا

indentation of nested

ترحيل التداخل

Input procedure ... Giving

أجزاء مدخلات ... معطيا

Input procedure.. output procedure

أجزاء مدخلات .. أجزاء مخرجات

Imperative statements

عبارات أمرية

Implicit redefinition

إعادة تعريف ضمنية

Indentation

ترحيل

Independent item

عنصر مستقل

Index

فهرس

Index records

سجلات فهرس

index required

مطلوب فهرس

INDEXED BY

مفهرس بواسطة

Indexed files

ملفات مفهرسة

indexing

فهرسة

Infinite loop

لوحة لا نهائية

Initialization

وضع قيم ابتدائية

Input

مدخلات

Input devices

وحدات مدخلات

INPUT-OUTPUT SECTION

قسم المدخلات والمخرجات

Input-output symbol

رمز مدخلات ومخرجات

INPUT PROCEDURE

إجراء مدخلات

Input-process-output cycle

لوحة مدخلات - عمليات - مخرجات

inserting new entry insources

إسخال محتوى جديد على التتابع

Insertion sort

ترتيب الإسخال

Insertion

إسخال

INSTALLATION	مؤسسة
Interblock gap	فراغ ما بين المجموعات
Intelligent	ذكية
Intermediate key	مفتاح متوسط
Invalid	غير صحيح
INTO	في
IRG (interrecord gap)	فراغ ما بين السجلات
Iteration structure	هيكل التكرار

J

Job control language (JCL)	لغة تحكم العمل
Julian date	تاريخ برقم اليوم
JUSTIFIED	مضبوط

K

Key	مفتاح
Key field	حقل رئيسي (مفتاح)
Key-to-disk	من لوحة المفاتيح للقرص
Keypunch	التتقيط باللوحة
Kilobyte (KB)	كيلوبايت

L

LABEL RECORDS	سجلات العنوان
Leading	رائدة
Length:	الطول
LESS THAN	أقل من
Level 01	المستوى 01
Level 77	المستوى 77
Level 88	المستوى 88
Level numbers	أرقام المستويات
LINAGE	خطي
Line image areas	منطقة صورة السطر
Linage	خطي
Linear IF	خطي
Linear table search	بحث جدول خطي

LINES AT BOTTOM	أسطر في المؤخرة
LINES AT TOP	أسطر في المقدمة
Literal:	ثابت
LOCK, WITH	اغلق بـ
Loading with read	تحميل بعبارة اقرأ
Logic error	خطأ منطقي
Loading	تحميل
Loading with value	تحميل بالقيمة
Logical deletion	حذف منطقي
Logical design	تصميم منطقي
Logical expressions, evaluation of	تعابير منطقية ، تقويمها
Logical operator	مؤثرات منطقية
Logical page	صفحة منطقية
Logical record	سجل منطقي
logical record copies	نسخ سجلات منطقية
Loop	دورة
LOW-VALUES	أقل قيمة في تسلسل التتابع

M

Machine language	لغة الآلة
Magnetic disk (see Disk)	قرص ممغنط
Magnetic tape	شريط ممغنط
Main program loop (logic design)	دورة برنامج رئيسية (تصميم منطقي)
Maintenance programming	برمجة صيانته
Major key	مفتاح رئيسي
Megabyte (MB)	ميغابايت
Memory dump	نفايا ذاكرة
MERGE,	الدمج
Merging	دمج
Merging unsorted files	دمج ملفات غير مرتبة
Microprocessor	ميكروبروسسور - مشغل دقيق
Millisecond (ms)	ميلي ثانية
Minor key	مفتاح أصغر
Minus sign:	إشارة سالب
MIPS (millions instructions per second)	مليون من التعليمات في الثانية

Misuse	استخدام خاطيء
MOVE	انقل
Moving variable - length	نقل طول متغير
Multifile:	متعدد الملفات
Multikey sort	ترتيب متعدد المفاتيح
MULTIPLY	اضرب
Multivolume file	ملف متعدد الحجم

N

Name	اسم
Nanosecond (ns)	نانو ثانية
NAMED	مسمى
NATIVE	تحديد تسلسل التتابع التقليدي
NEGATIVE	سالب
Nested PERFORM	نقذ متداخلة
NEXT SENTENCE	الجملة التالية
nested IF	إذا متداخلة
New master file	ملف رئيسي جديد
Nine (9, picture character)	9 (رمز صورة)
Nonnumeric	غير عدد
Nonprinter	عددية
Nonfatal logic errors	أخطاء منطقية غير جسيمة
Numeric	غير طابع
NOT	ليس
Nonprinter	غير طابع
NUMERIC	عددي
Numeric data	بيانات عددية

O

OBJECT-COMPUTER	كمبيوتر الهدف
Object program	برنامج الهدف
OCCURS	يحدث
Old master file	ملف رئيسي قديم
ON proced - name	على اسم اجراءات
ON	عند

ON All Procedures	على كل الإجراءات
ON All Prefrnces	على كل الاشارات
ON File - name	على اسم الملف
ON - identifier	على معرف
On procedure - name	على اسم إجراءات
ON SIZE ERROR	عند حدوث خطأ في الحجم
One - dimensional	نوبعد واحد
OPEN	الفتح
OUPUT	مخرجات
Operation OF	عملية
Operating system	نظام تشغيل
Operational sign	اشارة تشغيل
Operator's console	نهاية طرفية لمشغل الكمبيوتر
OR	أو
ORGANIZATION IS	التنظيم يكون
Output	مخرجات
OUTPUT PROCEDURE	إجراء مخرجات
Overflow	سريان زائد

P

P (editing character)	P (رمز تنقيح)
Packed-decimal data	بيانات عشرية مضغوطة
Padding	ملء فراغات أو أصفار
Page body	جسم الصفحة
Page overflow	سريان زائد الصفحة
Pagination without LINAGE	عمل صفحات بدون خطية
Paragraph	مقطع
Parallel tables	جداول متوازية
Parentheses (COMPUTE)	أقواس (احسب)
PERFORM	نفذ
Perform ... Varying	نفذ ... مغيراً
Period (editing character)	نقطة (رمز تنقيح)
Perioduse of,	نقطة ، استخداما
Physical deletion	حذف واقعى
Physical record	سجل واقعى (طبيعى)

PIC (see PICTURE)	صورة
PICTURE	صورة
Picture characters	رموز الصورة
Plus sign:	إشارة موجب
POSITIVE	موجب
Predefined process symbol	رمز تشغيل سابق التعريف
Priming read	قراءة أولية
Printers	طابعات
Procedure Division Structure	هيكل جزء الإجراءات
PROCEDURE DIVISION	جزء الإجراءات
Process symbol	رمز عملية
PROGRAM COLLATING SEQUENCE	تسلسل تتابع البرنامج
Program constants	ثوابت البرنامج
Program development:	إعداد البرنامج
problem definition	تعريف المشكلة
PROGRAM-ID	تعريف البرنامج
Program logic	منطق البرنامج
Program stubs	أجزاء برنامج وهمية
Program testing	اختبار البرنامج
Program trace	تتبع البرنامج
Programmer-defined names	أسماء يعرفها المبرمج
Prompts	ملقنات
Pseudocode	شفرة شبيهة
Pseudocode equivalent	مكافئ الشفرة الشبيهة

Q

Qualification	تأهيل
Quoted string	سلسلة موضوعة بين علامتي تنصيص
QUOTES	علامات تنصيص

R

Random	عشوائي
Random processing	تشغيل عشوائي
READ	اقرأ
Read/write heads	رؤس قراءة وكتابة

Receiving field	حقل مستقبل
Record	سجل
Record codes	شفرات (رموز) سجل
RECORD CONTRAINS	السجل يحتوى
RECORD CONTAINS 0 CHARACTERS	السجل يحتوى صفر رمزاً
Record deletion	حذف سجل
Record description	وصف سجل
Record descriptor word	كلمة واصف سجل
Record insertion	إدخال سجل
Record names	أسماء سجلات
Record number	رقم سجل
Record updating	تجديد سجل
Recursion	إعادة ذاتيه
REDEFINES	يعيد تعريف
Reel	بكرة
Relation condition	شرط علاقى
Relative	نسبى
Relative indexing	فهرسة نسبية
RELEASE	يذبح
REMAINDER	الباقى
REMOVAL, CLOSE FOR	إزالة , قريبا لـ
Repetition factor	معامل تكرار
Replacing relations	علاقات إحلال
RESERVE	محجوز
Reserved words,	كلمات محجوزة
RESET TRACE	إعادة إعداد التتبع
RETURN	أعد
Retrieval time	وقت استرجاع
Reversed	عكس
REWIND , WITH NO	إعادة لف , برقم
Restrictions on perform	قيود على نفذ
REWRITE	إعادة كتابة
Rotational delay	تأخير اللوران

ROUNDED

Rounding

Rule for nesting

مقرباً

التقريب

قاعدة إعادة ذاتية

S

S (picture character)

SD

SEARCH

SEARCH ALL,

SECTION

Section header

SECURITY

SELECT

Selction structure,

Self-documenting language

Sending field

Sentences,

Sequence check

Sequence structure

Sequential

Sequential file updating

Sequential processing

Sequential table search

SET:

Severe error

SIGN

Sign condition

Sign control editing

Simple

Simple conditions

SIZE ERROR

SKIP 1, SKIP 2 SKEP 3

Slack bytes

Salsh (editing character)

SORT

S (رمز صورة)

وصف ملف الترتيب

ابحث

ابحث كل

قسم

عنوان قسم

أمن

اختر

هيكل اختيار

لغة توثيق ذاتي

حقل راسل

جمل

اختبار تتابع

هيكل تتابع

تتابعي

تجديد ملف تتابعي

تشغيل تتابعي

بحث جداول تتابعيا

ضع

خطأ جسيم

إشارة

شرط اشارة

تنقيح تحكم الإشارة

بسيطة

شروط بسيطة

خطأ الحجم

اقفز ١، اقفز ٢ ، اقفز ٣

بايت راكدة

شرطة مائلة (رمز تنقيح)

رتب

Sort file	ملف ترتيب
Sort keys,	مفاتيح الترتيب
Sort/merge utility	تسهيلات الدمج والترتيب
SOURCE-COMPUTER	كمبيوتر المصدر
Source documents	وثائق المصدر
Source program	برنامج المصدر
SPACES	فراغات
Special characters	رموز خاصة
SPECIAL-NAMES	أسماء خاصة
STANDARD - 1	نمطى - ١
Statements	عبارات
STOP literal	قف ثابت
STOP RUN	أوقف التنفيذ
Stored PROGRAM, EXECUTION OF	برنامج مخزن ، تنفيذه
Structure	هيكل
Structure chart	خريطة هيكل
Structured flowchart	خريطة مسار مرتبة
Structured period	نقطة مرتبة
Stubs	أجزاء وهمية
Subfield	حقل جزئى
Subscribing	عمل أدلة
Subscripts	أدلة أو دلائل
Subscript conversion	تحويل الدليل
SUBTRACT	اطرح
Switches	مفاتيح
Symbolic dump	نفايا رمزية
SYNC (see SYNCHRONIZED)	متزامن
SYNCHRONIZED	متزامن
Syntax:	تكوين
Syntax error	خطأ تكوينى
Syntax notation	ترميز تكوينى
SYSIN	اسم خاص لوحد مدخلات
SYSOUT	اسم خاص لوحد مخرجات
Systems flowchart	خريطة مسار النظام

T

Table	جدول
two-dimensional summing entries	تجميع محتويات ذات بعدين
Table look-up:	فحص جدول
Table sort	ترتيب جدول
Subscript conversion	تحويل دليل
Tape (magnetic)	شريط (مغناطيس)
Terminal:	نهاية طرفية
Terminal Symbol	رمز نهاية طرفية
Test data	بيانات اختبارية
Testing	اختبار
THRU OPTION	جزء خلال
TIME	وقت
TIMES	مرة إلى
THRU TO	خلال
Top-down, modular design	تصميم أجزاء من القمة إلى القاعدة
Top margin	هامش علوي
To	لـ
Trace	تتبع
Trailer label	عنوان خلفي
Trailing	متأخرة
Transaction file	ملف عمليات جارية
Transaction register	مسجل عمليات جارية
Truncation	حذف
Tow level	مستويان
Two-dimensional tables	جداول ذات بعدين
Typing rules	قواعد الكتابة
Top - down coding and design	كتابة شفرة واختبار من القمة الى القاعدة

U

Unary minus	سالبة أحادي
Undefined,	غير معرف
Unit	وحدة
Unit record devices	وحدات سجل وحدة

Until in cobol	حتى في كويل
Until	حتى
Unsorted files	ملفات غير مرتبة
Update in place	تجديد في نفس المكان
Up By..	لأعلى بـ ..
USAGE	استخدام
Use in if	مستخدم في إذا
Use in perform	مستخدم في نفذ
USE FOR DEBUGGING	يستخدم في التصحيح
USING	مستخدم في
Utility programs	برامج تسهيلات

V

V (picture character)	V (رمز صورة)
Validation	التأكد من الصحة
VALUE	قيمة
versus MOVE	عكس نقل
Variable length	متغيرة الطول
Variable-length records	سجلات متغيرة الطول
Variable-length tables	جداول متغيرة الطول
Varying	مع تغيير
Verbs	أفعال
Versus Value	ضد القيمة
Vertical numbering	ترقيم رأسى
Virtual storage	تخزين افتراضى
Visual table of contents	جدول مرئى للمحتويات
Volume	حجم
Volume label	عنوان حجم
Volume switching	تغيير الحجم
Volume table of contents (VTOC)	حجم جدول المحتويات

W

Warning	تحذير
WHEN	عندما
WITH DEBUGGING MODE	مع حالة التصحيح

WORKING-STORAGE

مخزن العمل

WRITE

اكتب

X

X (picture character)

X (رمز صورة)

Z

Z (picture character)

Z (رمز صورة)

ZERO

صفر

Zero (editing character)

صفر (رمز صورة)

PROGRAMMING WITH STRUCTURED COBOL (Schaum)

صدر أيضاً للنشر

فى

الحاسبات

- * البرمجة بلغة الباسكال كيلر
- * الدوائر المتكاملة الرقمية والحاسبات ولارد
- * المجهزات والحاسبات الدقيقة لطلبة الهندسة والفنيين ولارد
- * الدوائر المنطقية واستخدامات المجهزات الدقيقة موريس
- * المدخل لعلم الحاسبات بارتنى
- * البرمجة بلغة الباسكال (شوم) جوتفريد
- * البرمجة بلغة البيسك (شوم) جوتفريد
- * البرمجة بلغة الفورتران (شوم) ليبشتز
- * الرياضيات الاساسية للحاسب (شوم) ليبشتز
- * كويك بيسك ناميروف
- * موسوعة مصطلحات الكمبيوتر محمود الشريف

الناشر

الدار الدولية للنشر والتوزيع

٢٨ ش الإهرام - روكسى - مصر الجديدة

ص.ب : ٥٥٩٩ هليوبوليس غرب - القاهرة

ت: ٢٥٨٢٨٨٧

تلکس : ٢٠٠٧٠ PBCRB UN

فاکس : ٢٩١٨٠٥٩ / ٢٠٢٠

ISBN : 977 - 5107 - 26 - 1